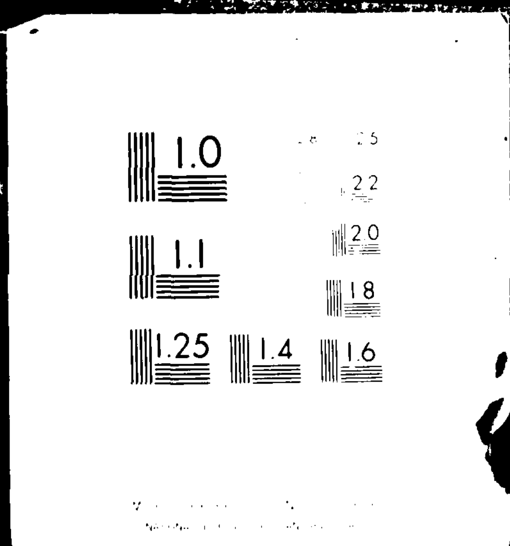
AD-A112 564 STANFORD UNIV CA COMPUTER SYSTEMS LAB F/G 17/2
SIMULATION OF MESSAGE-BASED PRIORITY FUNCTIONS IN CARRIER SENSE--ETC(U)
JUN 81 N GONZALEZ-CAWLEY, F A TOBAGI DAAG29-79-C-0138
UNCLASSIFIED TR-213 ARO-16783.2-A-EL NL

1 OF 2
AD A
112564

1 OF 2

AD A 112564



2. The p-persistent P-CSMA Protocol

COMPUTER SYSTEMS LABORATORY

DEPARTMENTS OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
STANFORD UNIVERSITY STANFORD, CA 94305

11

AD A112564

SIMULATION OF MESSAGE-BASED PRIORITY FUNCTIONS IN CARRIER SENSE MULTIACCESS/BROADCAST SYSTEMS

by

Noel Gonzalez-Cawley and Fouad A. Tobagi

1 June 1981

Technical Report No. 213

This research was supported by the U.S. Army, CECOM, Fort Monmouth, New Jersey, under Army Research Office Contract No. DAAG 29-79-C-0138

DTIC FILE COPY

82 03 24 025

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: Technical Report No. 213 — 2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): SIMULATION OF MESSAGE-BASED PRIORITY FUNCTIONS IN CARRIER SENSE MULTIACCESS/BROADCAST SYSTEMS	5. TYPE OF REPORT & PERIOD COVERED: Technical Report 6. PERFORMING ORG. REPORT NUMBER: 213
7. AUTHOR(s): Noel Gonzalez-Cawley and Fouad A. Tobagi	8. CONTRACT OR GRANT NUMBER(s): DAAG 29-79-C-0138
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Stanford Eelctronics Laboratories, Stanford University, Stanford, CA 94305	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: U.S. Army Research Office, PO Box 12211, Research Triangle Park, NC 27709	12. REPORT DATE: 1 June 1981 13. NUMBER OF PAGES: 90
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified 15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Reproduction in whole or in part is permitted for any purpose of the U.S. Government.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

THE VIEW, OPINIONS, AND/OR FINDINGS CONTAINED IN THIS REPORT ARE TH... AN OFFICIAL ... CISION, UNLESS SO DESIGNATED BY OTHER DOCUMENTATION.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Packet communication systems, multiaccess protocols, broadcast transmission, carrier sense, collision detection, priority functions, ETHERNET, packet radio networks.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The need for priority functions in Multiaccess Computer Communication Netwo ks arises from applications with restrictions on packet delay. One alternative is to have a dedicated channel supporting very low loads to insure small delays; another alternative is to provide some priority mechanism by which packets that are time constrained can have priority over those which are not.

One such mechanism, called Prioritized Carrier Sense Multiple Access (P-CSMA), recently proposed and analyzed by Tobagi [10-12], is studied here using simulation. The objective of this work is to extend the results obtained by the stochastic analysis and evaluate more completely the performance of P-CSMA. Three variations of the operation of the protocol are investigated, namely: nonpreemptive, semipreemptive and preemptive disciplines. In particular, we study the effect on average packet delay, packet loss and the variance of delay of several system parameters that prove to be interesting, such as: the number of stations, the number of buffers, the preemption discipline, etc.

It is shown that priority functions indeed reduce packet delay, delay variance and packet loss for the high priority class.

In addition, we investigate the problem of transmitting packetized voice on CSMA-CD local networks. Assuming P-CSMA and only voice traffic on the channel, we define network performance as the maximum number of voice sources accommodated for a given maximum delay requirement and a tolerable loss rate. We study the effect on this performance of various system parameters such as channel bandwidth, vocoder rate, delay requirement and packet loss rate.

SIMULATION OF

MESSAGE-BASED PRIORITY FUNCTIONS IN CARRIER

SENSE MULTIACCESS/BROADCAST SYSTEMS*

by

Noel Gonzalez-Cawley and Fouad A. Tobagi

1 June 1981

Technical Report No. 213

Computer Systems Laboratory
Stanford Electronics Laboratories
Depts. of Electrical Engineering & Computer Science
Stanford University
Stanford, CA 94305

form 50 per

A

This research was supported by the U.S. Army, CECOM, Fort Monmouth New Jersey, under Army Research Office Contract No. DAAG 29-79-C-0138.

SIMULATION OF

MESSAGE-BASED PRIORITY FUNCTIONS IN CARRIER

SENSE MULTIACCESS/BROADCAST SYSTEMS*

Noel Gonzalez-Cawley and Fouad A. Tobagi
Department of Electrical Engineering
Stanford University
Stanford, CA 94305

(415) 497-1708

TECHNICAL REPORT NO. 213

1 June 1981

ABSTRACT

The need for priority functions in Multiaccess Computer Communication Networks arises from applications with restrictions on packet delay. One alternative is to have a dedicated channel supporting very low loads to insure small delays; another alternative is to provide some priority mechanism by which packets that are time constrained can have priority over those which are not.

One such mechanism, called Prioritized Carrier Sense Multiple Access (P-CSMA), recently proposed and analyzed by Tobagi [10-12], is studied here using simulation. The objective of this work is to extend the results obtained by the stochastic analysis and evaluate more completely the performance of P-CSMA. Three variations of the operation of the protocol are investigated, namely: nonpreemptive, semipreemptive and preemptive disciplines. In particular, we study the effect on average packet delay, packet loss and the variance of delay of several system parameters that prove to be interesting, such as: the number of stations, the number of buffers, the preemption discipline, etc.

It is shown that priority functions indeed reduce packet delay, delay variance and packet loss for the high priority class.

In addition, we investigate the problem of transmitting packetized voice on CSMA-CD local networks. Assuming P-CSMA and only voice traffic on the channel, we define network performance as the maximum number of voice sources accommodated for a given maximum delay requirement and a tolerable loss rate. We study the effect on this performance of various system parameters such as channel bandwidth, vocoder rate, delay requirement and packet loss rate.

---

* This research was supported by the U.S. Army, CECOM, Fort Monmouth New Jersey, under Army Research Office Contract No. DAAG 29-79-C-0138.

KEYWORDS: packet communication systems, multiaccess protocols, broadcast transmission, carrier sense, collision detection, priority functions, ETHERNET, packet radio networks.

§1 Introduction

The low cost of computing systems have given rise to environments where, within a building or a small group of buildings, many different computers, minicomputers and intelligent microprocessor driven devices are used. The need to interconnect them in a uniform and reliable manner has led to the design of local area networks of different types. Packet radio technology also has advanced to the point where it is feasible to interconnect several computers through a radio channel. A local area network, or local network, is composed of two basic elements: (i) the transmission medium, generally twisted pair or coaxial cable, and (ii) the interface to the network from the computers or other devices, called the nodes of the network. This interface controls the mechanism for accessing the transmission medium as well as other levels of protocol.

The physical interconnection of the nodes in the network determine its topology. Several topologies exist for local area networks: (i) an unconstrained topology, (ii) the star topology, (iii) the ring topology, (iv) the bus topology.

(i) In an unconstrained topology a node can be connected to an arbitrary number of different nodes in the network. This topology presents the problem that there can be several paths, of different lengths, between any two nodes. A node receiving a packet that must be forwarded to its intended destination must make a decision regarding the links over which to retransmit it and this may place a heavy computational burden on the node.

(ii) The star topology consists of having each node connected to a central node through a unique link. The central node either has the capability to handle transmissions from all nodes simultaneously, or polls them one by one giving transmission right to only one of them at any one time. This topology is suitable for distributed computing systems, where a large mainframe computer talks to a large number of terminals, but has a reliability problem because the system depends on the correct operation of the central node.

(iii) In the ring topology, each node is connected to its two adjacent nodes only, in such a way that all nodes form a ring. Communication is achieved by sending packets unidirectionally through the ring and the computational burden placed on a node is that

it be able to recognize messages intended for it.

(iv) In the bus topology all nodes are connected to a common transmission medium and send their packets bidirectionally through the bus. Again a node must be able to recognize packets intended for it.

The bus is a passive element that poses no practical reliability problems. The problem introduced by the ring and bus topologies as well as with Packet Radio environments, is that of controling the flow of messages and the access to the channel in an organized and reliable manner. In these topologies there is no central node so they use distributed mechanisms to control the flow of information.

Several access control mechanisms have been developed and we discuss here some of them:

One such mechanism, used in ring networks, is the control token [5], where the system uses a predefined control token that is passed around the ring. A station wishing to send a packet waits for this contol token and when it sees it go by it retrieves it from the network and inserts its own message, after which it reinserts the control token again. Several provisions have been developed to deal with the loss or duplication of the control token. Another technique is the use of message slots [7], which are sequences of bits long enough to hold a full message. A message slot can be either full or empty. A station wishing to transmit waits until an empty slot comes by, marks it full and inserts its message. Of course, one node must initiate the slot pattern, so the system is not completely decentralized.

Bus topologies and radio environments also require a decentralized control strategy; we will discuss some of the techniques used shortly but first let us discuss two of the major types of control strategies that can be used in this kind of environment: fixed assignment and random access, with some techniques incorporating a little of each type[2,9].

In fixed assignment, each user is allocated a fraction of the total available bandwidth to be accessed exclusively by him when it wishes to transmit, clearly it is wasted when this user is idle. One major advantage of these types of schemes is that they are free of conflicts,

as each user holds exclusive access to a fraction of the available bandwidth; one major disadvantage, on the other hand, is that a user which requires more bandwidth can not use it even if other users are idle, therefore wasting channel capacity.

In *random* access techniques there is very little control, and in some mechanisms used, there is no control at all. In the ALOHA system [1] for example, which operates in a radio environment, a station with a message to transmit simply transmits it; there is the possibility that a station transmits while another one is also transmitting producing a conflict on the channel, we call such a conflict a *collision*. After a packet has collided, which is detected by the absence of an acknowledgement from the destination, it is retransmitted again after a random retransmission delay. Collisions lead to a serious degradation of the channel when it is heavily loaded placing a limit of 18% on the channel capacity for this system. One way to improve this performance is to consider the time slotted, with the slot size equal to the packet length. Stations wishing to transmit do so at the beginning of a slot, therefore, if packets collide, they collide completely, causing the channel capacity to increase to 36%.

In local environments where the transmission medium is coaxial cable or twisted pair, for example, the feature of collision detection is available, i.e., a station can detect when there is a conflict on the channel and abort its ongoing transmission immediately and reschedule its transmission for a later time, based on a random retransmission delay. This feature greatly improves the channel capacity.

One other feature that is incorporated in broadcast networks is that of sensing the channel before transmitting. A station can detect the presence of carrier on the channel and refrain from starting a transmission as long as this carrier is present, thus avoiding a collision, which can only occur now at the beginning of a transmission. In radio environments only carrier sensing is available, collision detection can not be incorporated in these environments. These two features are incorporated into the ETHERNET[6], a local communication network which uses carrier sensing, a bus topology and coaxial cable as the transmission medium. This packet broadcast network has proved very successfull.

1. **Introduction**

Control techniques which sense the channel before transmitting are called Carrier Sense *Multiple Access -CSMA- techniques*[3], and two very important versions of it are the nonpersistent and p-persitent CSMA protocols. They both present the advantage of reducing the chance of a collision at the beginning of a transmission by having stations schedule the beginning of its transmissions acording to some scheduling policy, trying to spread in time these starting points.

In nonpersistent CSMA a station with a message ready for transmition operates as follows:

a. if the channel is sensed idle the station transmits its packet.

b. if the channel is sensed busy a station reschedules its transmission to some later point in time according to a retransmission delay distribution, at which point repeats the algorithm.

In the p-persistent protocol a station with a ready message operates as follows:

a. if the channel is sensed idle the station transmits its packet.

b. if the channel is sensed busy, the station waits until it becomes idle and then with probability p transmits the packet, and with probability $1-p$ delays the transmission by τ seconds, where τ is the end to end propagation delay of the channel. If at this new point in time the channel is sensed idle, the station repeats the process. If another station started transmitting, the station may operate in one of two fashions: it may reschedule the transmission of the packet according to a retransmission delay distribution or repeats step **b**.

If collision detection is available we call the protocol Carrier Sense Multiple Access with Collision Detection, CSMA-CD [8].

One advantage of using CSMA is the high channel utilization that can be achieved, which can be further increased if collision detection is available; the delay, on the other hand, is by no means fixed due to the random nature of the access algorithm and it has been shown to grow unbounded as the total channel throughput aproaches channel capacity [8]. The protocol is independent of the packet size, therefore packets of different lengths

can be sent through the network. If we are multiplexing several applications on the same channel it is not unusual to find long packets and short ones sharing the same channel. Long packets are typical of file transfers, for example, while short packets are generally used in interactive applications, voice transmission, process control, sensors data, flow regulating control packets, etc. While it is true that these applications typically do not place a heavy load on the channel it is also true that many of these applications have a common time constraint. On the other hand, typically the transmission of long packets can tolerate large, yet reasonable, delays. Unfortunately in CSMA the degradation of the channel as it aproaches its capacity makes no discrimination among packets of different applications, and we cannot reliably guarantee a small delay if the total load placed on the channel is high. The introduction of priority functions into the protocol will allow packets with time constraints to have priority over those that don't have such constraints, allowing small delays for high priority classes as long as the load placed on the channel by these classes is itself small, independently of the load exercised by low priority classes.

One such scheme, called Prioritized Carrier Sense Multiple Access, P-CSMA, has been proposed by Tobagi[12], and analytical work has shown some very promising results.

The system proposed incorporates priority functions in the p-persistent CSMA protocol. The difficulty of the analytic work however, has prevented the evaluation of the effect of several system parameters that prove to be interesting, as well as variations of the operation of the protocol. The present work attempts to overcome, via simulation, the limitations of the analysis in evaluating the effect that several parameters have on the system's performance. At the same time it verifies the results obtained in the analysis, which further validates the simulation model itself.

In Section 2 we describe the protocol as originally proposed by Tobagi[12]. In Section 3 we describe the abstract model used to represent the environment and define the performance measures. In Section 4 we present the simulation program. In Section 5 we discuss the numerical results obtained from the simulation. In Section 6 we study voice transmission on local networks and finally in Section 7 we present some conclusions and we suggest some ideas for future work.

§2 The p-persistent P-CSMA Protocol

If we are going to introduce priority *functions in a Computer Communication* Network, we require that the performance of the scheme as seen by messages of a given priority class be insensitive to the load exercised on the channel by lower priority classes. Increasing loads from lower classes should not degrade the performance of higher priority classes. Also, several messages of the same priority class may be simultaneously present in the system and should be able to contend for the channel with equal right (fairness within each priority class). Obviously the scheme must be robust, i.e., its proper operation and performance should be insensitive to errors in status information. It is also desirable that the overhead required to implement the priority scheme, and the amount of control information exchanged among the contending users, as required by the scheme, be kept minimal.

Several aspects of the operation of a protocol must be addressed in order to establish priority functions in a distributed enviroment:

Even though this protocol does not require synchronization among all users, there has to be a consensus on the *time epoch at which priorities should* be assessed,a mechanism for reaching a consensus regarding the highest priority class present in the system. As well as provide the means to assign, in a fair manner, the channel to the various ready users within a class. The P-CSMA protocol resolves the first two problems by the use of reservation bursts and carrier sense and the third one by using the *p*-persistent CSMA protocol.

We present here a detailed description of the *p* persistent P-CSMA protocol as proposed by Tobagi[12]:

2.1 Basic Mechanism for Priority Assessment (a nonpreemptive discipline)

With the broadcast nature of transmission, users can monitor the activity on the channel at all times. The assessment of the highest priority class with ready messages is

done at the end of each transmission period, whether successful or not; i.e., every time the carrier on the channel goes idle. *When* detected at a user, end of carrier (EOC) establishes a time reference for that user. Following EOC, the channel *time* is *considered* to be slotted with the size of a slot (referred to as *reservation slot*) equal to $2\tau + \gamma$, where γ is the period of time of the shortest burst of unmodulated carrier which can be reliably detected. The priority of a user *at any time* is the highest priority class with messages present in its queue.

Let h denote an arbitrary user, and $t_e(h)$ denote the time of end of carrier at user h. Let $r(h)$ *denote* the priority level of user h at time $t_e(h)$. The priority resolution algorithm consists of having user h operate as follows:

(i) If, following $t_e(h)$, carrier is detected in reservation slot i, with $i < r(h)$, (thus meaning that some user(s) has priority i higher than $r(h)$ and access rights may be *granted* to class i), then user h awaits the following end of carrier (at the end of the next transmission period) at which time it reevaluates its priority and repeats the algorithm.

(ii) If no carrier is detected prior to the j^{th} reservation-slot, where $j = r(h)$, then user h transmits a short burst of unmodulated carrier of duration γ at the beginning of reservation-slot j (thus reserving *channel* access to priority class $r(h)$) and, immediately following this reservation-slot, operates according to the p-*persistent* CSMA protocol. That is, it senses *the channel* and: a) if the *channel* is *sensed* idle, then with some probability p it transmits the message, and with probability $1-p$ it delays action by τ seconds and repeats the CSMA procedure; b) if the channel is sensed busy, then the user awaits the next EOC and reevaluates its priority level and repeats the entire algorithm; c) if, during the time that channel access is granted to class $r(h)$, some user h' generates a (*new*) message of the same priority level, then h' transmits its message with probability one, provided that the channel is sensed idle. If, however, the channel is sensed busy at the message generation time, then h' awaits EOC, reevaluates its priority level and executes the algorithm. (Thus when a message is *generated, the user undertakes immediate first transmission* provided that the channel is idle and channel access is granted to the priority class corresponding to the newly generated message.)

(iii) If, following EOC, no reservation burst is detected for K consecutive reservation-slots, where K is the total number of priority classes available in the system, then the channel becomes free to be accessed by all users *regardless* of their priority, until a new EOC is detected.

Thus, by the means of short burst reservations following EOC, the highest non-empty priority class is granted exclusive access right, and messages within that class can access the channel according to p-persistent CSMA. Note that the above algorithm corresponds to a *nonpreemptive* discipline, since a user which has been denied access does not reevaluate its priority until the next EOC. However, note that by assessing the highest priority level at the end of each tranmission period, whether *successful or not*, the scheme allows higher priority messages to regain the access right without incurring substantial delays.

The scheme is robust since no precise information regarding the demand placed on the channel is exchanged among the users. Information regarding the existing classes of priority is implied *from the position of the burst of* unmodulated carrier following EOC. Note also that there is no need to synchronize all users to a universal time reference. By choosing the reservation-slot size to be $2\tau + \gamma$ we guarantee that a burst *emitted* by a transceiver in its k^{th} reservation-slot is received within the k^{th} reservation-slot of all other hosts.

2.2 Preemptive P-CSMA

Consider that after the reservation process has taken place, the channel has been assigned to class j. Assume that before a transmission takes place a message of level i, $i < j$, is generated at some host h. The nonpreemptive scheme dictates that host h awaits the next time reference before it can ascertain its (higher) level i. The *semi preemptive* scheme is one which allows host h to preempt access right to class j, as long as no transmission from class j has yet taken place, by simply transmitting the message (the transmission starting during the idle time representing CAP). If the generation of the message of level

i takes place after a transmission period is initiated, then host h waits until end of carrier is detected. Both nonpreemptive and semi-preemptive schemes are applicable whether collision detection is in effect or not.

A *fully preemptive* P-CSMA scheme is also defined in which a host with a newly generated packet may also preempt an on-going transmission of a lower priority level by intentionally causing a collision. Clearly this scheme is only appropriate if collision detection is in effect! It can offer some benefit if lower priority classes have long messages. One may also envision a partial preemption scheme whereby an on-going transmission is preempted only if the already elapsed transmission time has not exceeded some fraction of the total transmission time, where the packet transmission time is assumed to be known as is the case with fixed size packets.

§3 The Model

We describe in this section the model used in the study of the P-CSMA protocol.

The protocol does not require that all stations be synchronized to a common clock, but rather that all stations operate according to the time epoch of the system, which is determined by the events *on the channel*. In other words, we require only that all stations be aware that the system is in the priority assesment period, for example, which is achieved by the selection of the reservation slot size, but we do not require that this period start at the *same time* for all stations. This approach greatly simplifies the problem of synchronization among all users as no universal clock is required. Although the protocol does not require it, for performance evaluation we consider the system synchronized to a universal time clock. The time axis is then considered to be slotted and the slot size to be equal to τ, the end to end propagation delay; the reservation slot is of size 2τ, neglecting γ (the time needed to reliably detect a short burst of unmodulated carrier).

We consider a *broadcast network* with M stations, all of which monitor the activity on the channel and can detect two events: an ongoing transmission and the end of carrier. In a system with collision detection, when a station is involved in a collision, it will abort its transmission after a recovery time T_c.

A station has a fixed number of buffers per class and only packets of a given priority class can occupy the buffers assigned to that class. A packet that is generated by a station is placed on the buffer, or one of the *buffers, assigned* to that priority class. If when a packet is generated there are no buffers available for its particular priority class then this packet is considered lost.

In the nonpreemptive discipline, only those messages already in a station buffer when the Priority Assessment Period (PAP) begins will be able to place a reservation for their corresponding priority class. Once the channel has been granted to a given priority class, all messages that arrive during the contention period that correspond to different priority classes than the one with the current access right, will have to wait until a new PAP for a chance to ascertain their presence. Those packets of the current access class arriving

during the channel access period will transmit immediately, i.e., undertake immediate first transmission as mentioned in the description of the protocol. If a transmission on the channel is free of conflict, then it is assumed that the transmitted packet will arrive successfully at its destination and thus is deleted from the buffer; if a collision does occur the packet remains in the buffer and the station repeats the algorithm after the end of carrier detected when the collision was aborted (after a recovery time T_c).

3.1 *The Packet Generation Process*

So far we have mentioned how packets *contend for and gain access* to the channel but we have not described how they arrive at the system.

Neither the protocol nor the simulator restrict the type of arrival process that can be used in the network. For consistency with the analysis of the scheme[12], in this work we consider that at any given station, packets of priority class i are generated according to a geometric distribution with parameter σ_i and placed in the station's buffer for that particular class if there is room for it (otherwise it is considered lost). It is assumed that the arrival of packets coincides with the end of the slot in which it occurs, therefore no delay is incurred during the first slot. At any rate, this does not have a significant effect in the overall system performance.

3.2 Description of Variables

The system has several parameters that influence its performance, many of them have already been mentioned earlier; these are:

M_i: the number of stations in the system that generate packets of priority class i.

σ_i: the packet generation rate for priority class i.

B_i: the number of buffers per station for priority class i.

p_i: the p-persistent parameter used in the contention period by stations with messages of priority class i.

τ: the end to end propagation delay.

T_i: the packet length for packets of priority class i. Notice that a transmission period is one slot larger than the transmission time because it is assumed that it takes one slot for an EOC to propagate through the network and this slot is considered part of the transmission period; the unmodulated carrier during the PAP, however, does not need an extra slot because this slot is considered in the definition of the reservation slot.

T_c: the collision recovery time.

T_p: the time since the begining of a transmission in which preemption is allowed.

3.3 Measures of System Performance

There are basically four performance aspects of the system that we consider in this work:

(i) The average delay per packet D_i, defined as the time (in slots) spent in the system by a packet from the moment it arrives at the system until it is successfully received at its destination. Notice that this includes the transmission time of a packet, this places a lower bound on the packet delay of at least one transmission time.

(ii) The channel throughput S_i, which is the fraction of time that the channel carries successful transmissions. Channel throughput and packet delay are two closely related measures. The higher the throughput the larger the packet delay. This relationship between average packet delay and total channel throughput is referred to here as the throughput delay characteristic of the system.

(iii) The variance of delay Var(D_i), which is self-explanatory, and

(iv) Packet loss defined as the probability that a packet generated at any given slot will be lost because there is no place for it in the station's buffers. It is also the difference between the channel throughput and the total load exercised on the channel.

§4 The Simulator

The simulator used in this work was written in PASCAL for a DEC-20 system, running under the TOPS-20 operating system.

The simulation combines both event and time driven simulation techniques and consists of three major parts, each of which simulates one of the three periods of the protocol, namely: the priority assessment period, the channel access period and the transmission period. This modular organization of the program permits changes to be made in the mechanisms used in each period without affecting significantly the rest of the program.

One of the initial objectives of this work was to verify the analytical results that had been obtained so certain 'restrictions', not inherent to the simulation, were incorporated to follow as closely as possible the model considered in the analysis [12]. Examples of such restrictions are: considering the time axis slotted and considering packet arrivals to be coincident with the end of a slot.

4.1 Data Structures

The program comprises two data structures: a binary tree and a three dimensional array (1..M,1..K,1..B) where M is the number of stations, K is the number of priority classes and B is the number of buffers per station and priority class. The array holds packets queued at each station along with the information regarding the time at which the packet is supposed to have arrived or to be generated at that station. We keep this array always full; that is, at initialization packets are generated according to the geometrically distributed interarrival times with parameter σ_i for class C_i for all stations and all buffer positions at these stations. Furthermore, during execution of the simulation, whenever a packet is successfully transmitted and therefore removed from the system, a new packet is generated immediately, again using the geometric distribution of interarrival times. The binary tree holds the information of all the packets that are considered ready for transmission by the corresponding stations, i.e., the first packet of every queue in the system. The nodes hold information regarding the generation time of the packet, the station and the priority class

and are ordered in the tree according to their generation time in such a way that an inorder traversal of the tree [1] will visit the nodes in ascending order of generation time. This particular ordering of the data permits all searches for packets that might be transmitted to be of logarithmic order. When a packet is successfully transmitted the node which represents it is removed from the tree and a new node is inserted obtaining the appropriate generation time from the corresponding queue, and a new generation time inserted in the matrix of queues. Note that it is *not equivalent to say that in the real system the queues are all full.* Indeed, if the generation time of a packet in the array or in the message tree is higher than the current simulation time, then the packet is not yet present in the real system; therefore a station can have no messages to transmit, even though there is a node representing each of its priority classes in the tree.

4.2 Packet Generation and Transmission

When a packet is transmitted the node representing it is deleted from the message tree and a new packet of the same priority class is generated for the station which succeeded in the transmission. A generation time is obtained by taking as a reference time the last generation of a packet for that priority class and that station and adding to it a random number drawn from a geometric distribution with parameter σ_i. Notice that the reference time corresponds to the generation time of the packet just transmitted only in the case of *one buffer systems.* For systems with *more* than one buffer it is the generation time of the last packet in the corresponding queue. If the arrival time thus obtained is smaller than the current system clock the packet is considered lost and this new time reference is used to generate another packet. This process is repeated until a successful packet is generated and placed on the stations queue. The first packet in the queue is then inserted into the message tree.

4.3 System States and Immediate First Transmission

The system can be in either of two states: Empty or nonempty. The system is

empty when there are no messages present in the system. In a real system this will be the case when no station has a packet to transmit; in the context of the simulation this is the case when the generation times of all the packets in the system are greater than the current simulation time. A *nonempty system* corresponds to one in which one or more stations have packets ready for transmission; again in the context of the simulation, this corresponds to the existence of packets with generation times smaller than the simulation time. During the simulation, when there are no messages present on the system, the clock is advanced to the generation time of the leftmost node in the tree which corresponds to the first packet that will arrive, and an immediate first transmission is undertaken (indeed according to the protocol, the channel would be sensed idle at that time).

The tree is then searched for packets with arrival time equal to that of the leftmost node, indicating that it, too, will be transmited at the current time and will cause a collision. If no such packet exists then the clock is advanced by a period of time equal to the length of the transmission period of the packet being transmitted at which point the corresponding node is deleted from the message tree. Upon completion of successful transmission or of a collision, a Priority Assessment Period begins.

4.4 The Priority Assessment Period

We consider now the code corresponding to the priority assessment period.This part of the simulation is *time driven*. We begin by setting the priority level to the highest and we examine the message tree searching for packets of the current priority that can ascertain their presence. As mentioned earlier, only those packets generated before the beginning of the PAP can do so. If any packets are found a list of these packets is formed. The list is a linked list of pointers to the actual nodes of the tree. If no packets of the current access priority are present in the system, the priority level is decremented, the clock advanced two slots and the algorithm repeated until a nonempty class is found or all priority classes have been examined.

If all priorities have been examined and no packets have been found the system is

returned to the empty state and the simulation proceeds as we have already described.

If during the search of the message tree we find some packets of the current access class, then the channel is granted to this priority class and the clock is advanced two slots. In a real system packets would ascertain their presence by transmitting a short burst of unmodulated carrier in the slot corresponding to their priority class. Once the channel has been granted to a particular class no more searches are done and the system enters the channel access period.

4.5 *The Channel Access Period*

The part of the simulation that corresponds to the contention period is implemented as a time driven simulation. Let i be the priority class for which a list of messages has been formed, then for every packet in the list a uniformly distributed random number is drawn. If this number is less than p_i, the packet is marked for transmission in the next slot and a count is kept of the number of messages that will be transmitted. At this point we must search the message tree for any messages of the current access class arriving during the idle slot in which stations are sensing the channel. If one or more such packets are found it undertakes an immediate first transmission as described above (if more than one such new packet exists, then a collision will certainly result). If no incoming packet preempts the messages that are on the list the clock is advanced one slot and the count of the number of packets marked for transmission is examined. If this number is 0 the algorithm is repeated since no station decided to transmit. If the count is equal to 1, the marked packet is transmitted and the corresponding node deleted from the tree in the manner previously described. If there is more than one marked packet a collision will occur.

A collision is simulated by advancing the clock by the recovery time T_c, after which the system is returned to the Priority Assessment Period.

4.6 Statistics Collected

In order to evaluate system performance according to the criteria we have specified, several statistics are collected.

Packet delay is obviously an important quantity. It is calculated at the end of each transmission and is the difference between the current clock and the generation time of the packet. We accumulate this delay for each priority class and at the end the ratio of the total delay per class to the number of packets of that class that were transmitted gives the average delay per packet for each priority class.

We compute the square of the delay and accumulate it to obtain the variance of delay for each class.

To obtain the channel utilization per priority class we need only count the number of successful transmissions for each class. This number multiplied by the packet transmission time for that particular class divided by the simulation time gives the desired quantity.

Another pair of statistics collected for this work was the number of packets generated and the number of packets lost. This is important in several types of applications such as real time voice transmission and process control, where a packet cannot be recovered or generated again. It also gives a measure of the difference between the channel throughput and the load exercised by the stations.

Some other statistics were collected that were not actually used in the results presented here but helped in determining if the performance of the system was within reasonable limits; if this was not the case, then these statistics gave an indication of the parameter that was most likely to be causing the unexpected performance. An example of such numbers is the reservation and collision overhead which give an indication of the level of contention present in the system. If the number of collisions is very high, most likely the selection of p_1 is not appropriate and a smaller value may be needed. If on the other hand the length of the channel access period, which is also collected, is very large the parameter p_1 may be increased to improve system performance.

A word must be said about the selection of p_i. In general we wish to select p such that it minimizes the length of the contention period but maximizes the probability of success. Ideally, if we had a way to estimate the number of *users* with *ready* messages on the system, say N, a good selection could be $1/N$; in practice, however, N is difficult to estimate so we use a fixed number which depends on the number of users in the system, M, and the load they place on the channel. We have used for most of this work the parameter $p_i = 1/M_i$, which, if not optimum, keeps the number of collisions low and the channel access period small so that the results were close to optimum.

4.7 Other Disciplines Considered

In addition to the nonpreemptive discipline, we investigate here two other modes of operation not included in the analytical work, namely: semipreemptive and preemptive P-CSMA.

The only difference between these modes and the one already described is in terms of the protocol they perform after the end of Priority Assessment Period; the generation process and system configuration are the same.

In the semipreemptive mode, once the PAP has finished and a given priority class has acquired the right to access the channel, any message of higher priority that arrives to the system during the channel access period can preempt the current class when the channel is sensed idle, this is achieved by searching the message tree for packets that can preempt the current class, i.e., new arrivals of the same class or one of higher priority. All messages waiting will see a carrier on the channel and will back off until the next EOC when a new PAP will begin.

In the preemptive mode, we considered a fraction of the message, called T_p, in which preemption is allowed. Once a given class has been granted access to the channel and as long as the transmission time has not exceeded T_p, the specified limit, any message of a higher class can preempt the current priority class. Notice that if this preemption occurs during the channel access period the transmission of the high priority class will probably

be successful (unless another station has also decided to begin its transmission in the same slot). On the other hand, if the transmission of the low priority class has already started when the preemption occurs, the preempting packet will cause a collision whose EOC will cause a new PAP to begin, thus allowing the high priority message to gain access to the channel faster than in the other two modes of operation.

Several versions of the simulator are available that allow the study of systems with or without collision detection and with or without priority functions to evaluate the performance of the P-CSMA protocol in comparison with nonprioritized schemes.

§5 Numerical Results

We discuss in this section the numerical results obtained with the simulation and evaluate the performance of the scheme. Since the objective of this work is to study the effect of priority functions in multiaccess networks we shall in many cases compare the performance of prioritized systems to nonprioritized systems in oder to determine the overall effect produced by the priority functions. Since the approach taken will be that of comparing the three modes of operation among themselves, namely: the nonpreemptive, the semipreemptive and the preemptive mode, we divide this section into several subsections, one for each of the following aspects: Throughput-Delay Characteristics, Packet Loss, Variance of Delay. Finally we consider a special application case in which we simulate fixed packet interarrival times. We use this aproach to focus our attention on the basic differences between the modes of operation rather than consider all characteristics for one mode and then make a final overall comparison of the systems described.

Throughout this section we consider the unit of time to be $\tau = 10\mu sec$.

5.1 Throughput-Delay Characteristics

Ideally in a prioritized enviroment, the delay characteristics of a given priority class should be *insensitive* to the load exercised on the channel by all lower priority classes; in practice this is not exactly the case, due to the overhead incurred in the reservation process and the amount of time that a packet must wait until an ongoing transmission finishes, following which it can ascertain its presence in the system. We wish, however, to minimize the effect that the load of lower classes has on higher priority classes and, more important, insure that the delay for high priority classes is kept within tolerable limits regardless of that load. This is in contrast to other nonprioritized random access schemes in which the delay grows unbounded for all messages as the total throughput approaches channel capacity. We begin our discussion by considering a nonpreemptive system with two priority classes, C_1 and C_2, $M = 5$ stations, $B = 1$ buffer per station and packet lengths of $T_1 = 10$ and $T_2 = 100$ for the high and low priorities, respectively. The system

under consideration has collision detection and recovers from a collision with a recovery time $T_c = 2$ slots.

In Figure 1 we present the throughput delay curves for such a system, in which the parameters $p_i, i = 1,2$, used in the channel access period take on the different values $p_1 = p_2 = 0.1, 0.2, 0.5$. We observe that the delay for C_1 is dependent on the *total channel* throughput, *although the throughput for C_1 is kept constant* at $S_1 = 0.1$; but we can also note the fact that the delay for high priority messages is not growing unbounded as the total throughput $S_1 + S_2$ increases to reach saturation, contrary to the nonprioritized scheme, as it will be shown later. In fact the delay for class C_1 will *tend* to stabilize as the load for C_2 increases and more collisions start to occur within C_2, thus allowing C_1 messages to gain access to the channel faster, *given* that the collision detection is in effect. Note that the parameter p_i, in the case when M is small has rather little effect on the delay. In the sequel, for $M = 5$, we shall use the values $p_1 = p_2 = 0.2$, which gives near optimum results over the entire range of throughput (Note that this value is precisely $1/M$ as commented upon in the previous section).

In Figure 2 we examine a similar system, except that the number of stations has been increased to $M = 50$. It is easier here to observe the effect of *saturation on the* channel *and* the fact that the delay for the high priority class remains finite if not decreases as S_2 increases. Moreover, note that the effect of p_i is more important, and the choice of *optimum* p_i becomes more critical when M is larger. Here $p_2 = 0.1$ is the value for p_2 which gives us near optimum results. The load of C_1 on the channel is small and so is the probability of a collision; therefore the results are not too sensitive to p_1 *in this case* (*small* S_1) *and we use* $p_1 = 0.1$ *which still gives* good *performance* . We have, in effect, achieved a separation of the loads on the channel that allows us to obtain finite delays for C_1-messages as long as the load of this class stays below channel capacity.

In Figure 3, throughput-delay curves are shown for a five station system (hereafter refered to as small system) operating in the nonpreemptive mode, for $S_1 = 0.1$ and $S_1 = 0.2$. For comparison we include in Figure 3, the throughput-delay performance of CSMA-CD without priority functions. We notice that there is a difference in delay for the two

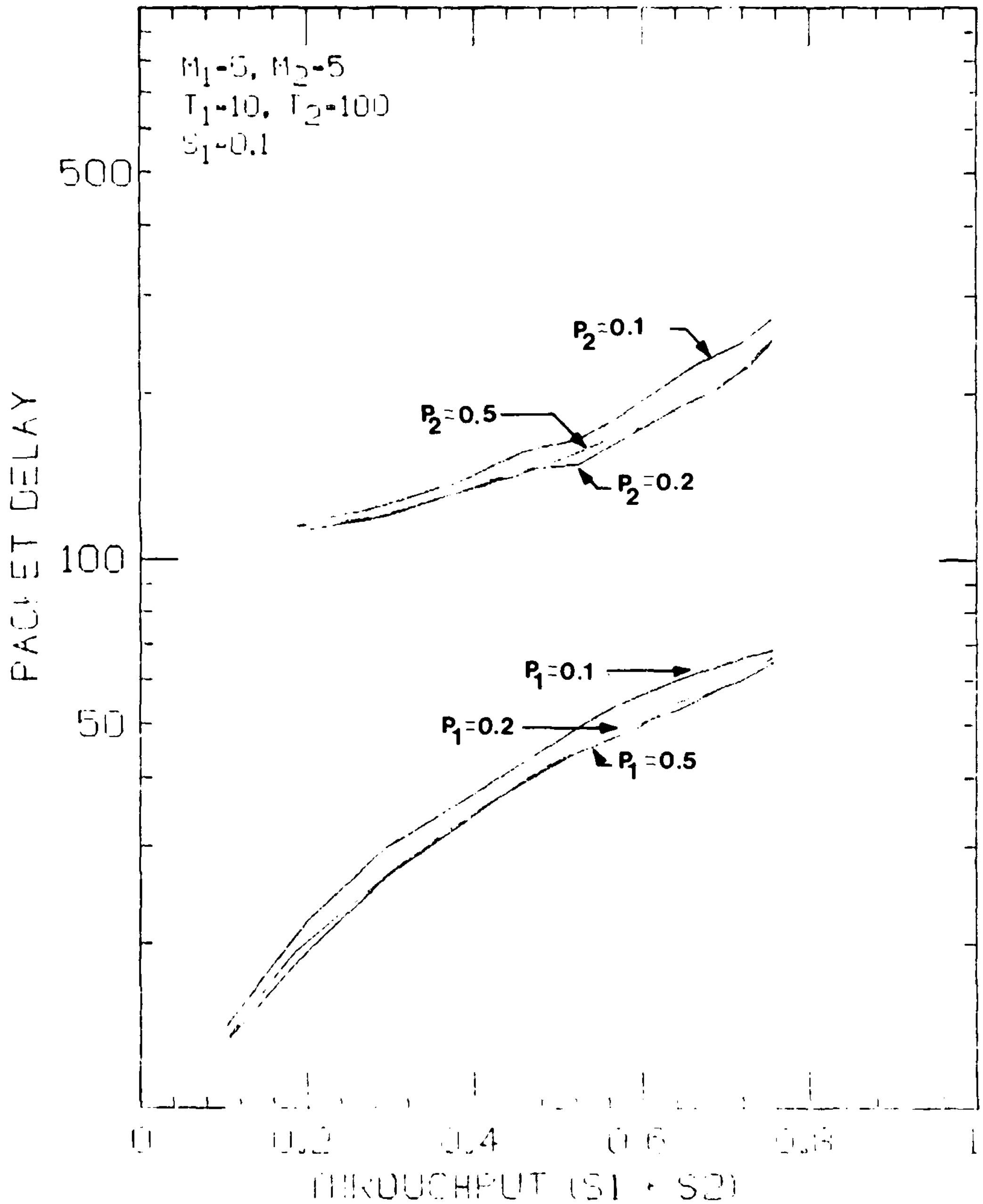


Figure 1. Average packet delay in nonpreemptive system with five stations and different *p*-persistent parameter values (0.1,0.2,0.5)

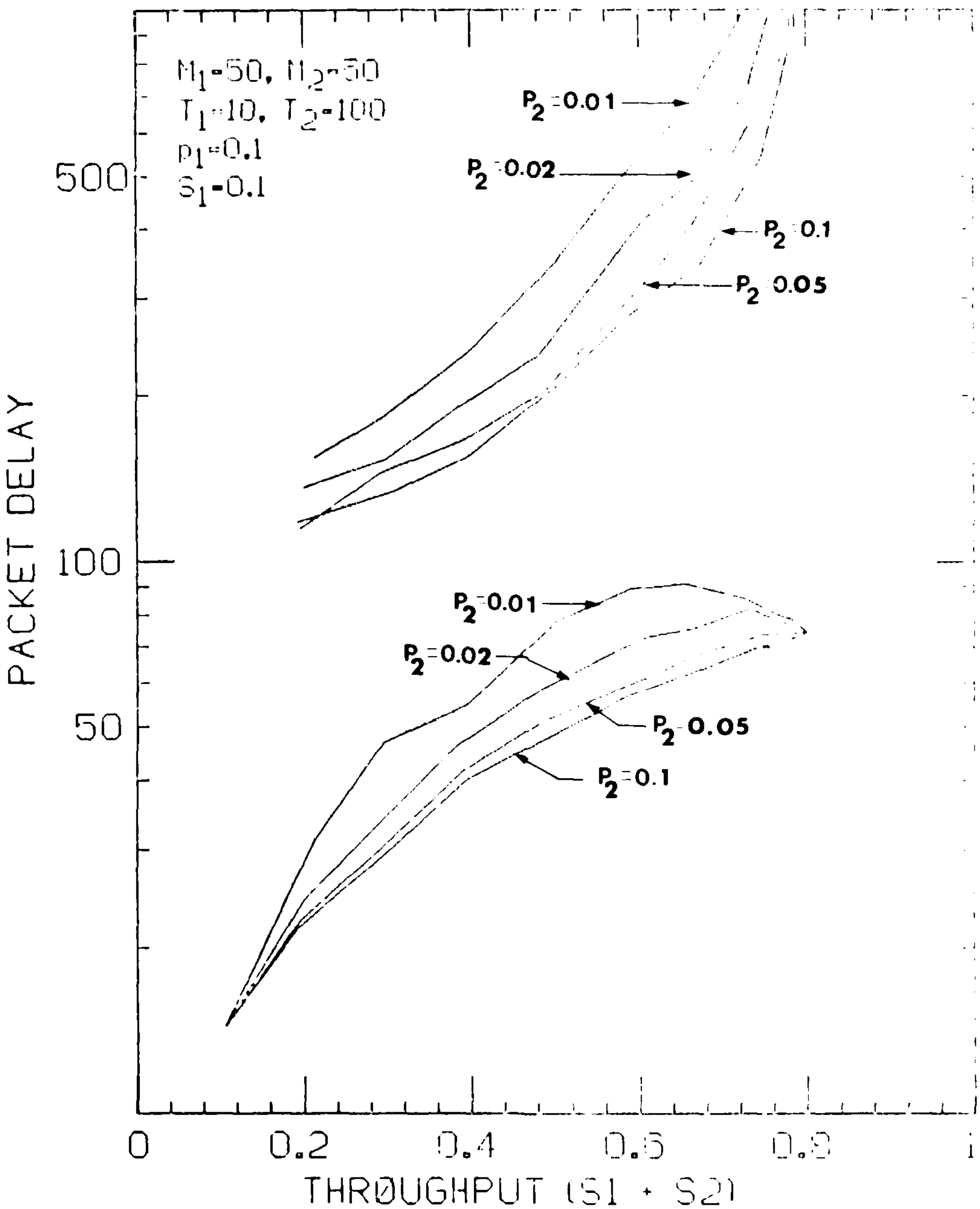


Figure 2. Average packet delay in nonpreemptive system with fifty stations and different *p*-persistent parameter values (0.01,0.02,0.05,0.1)

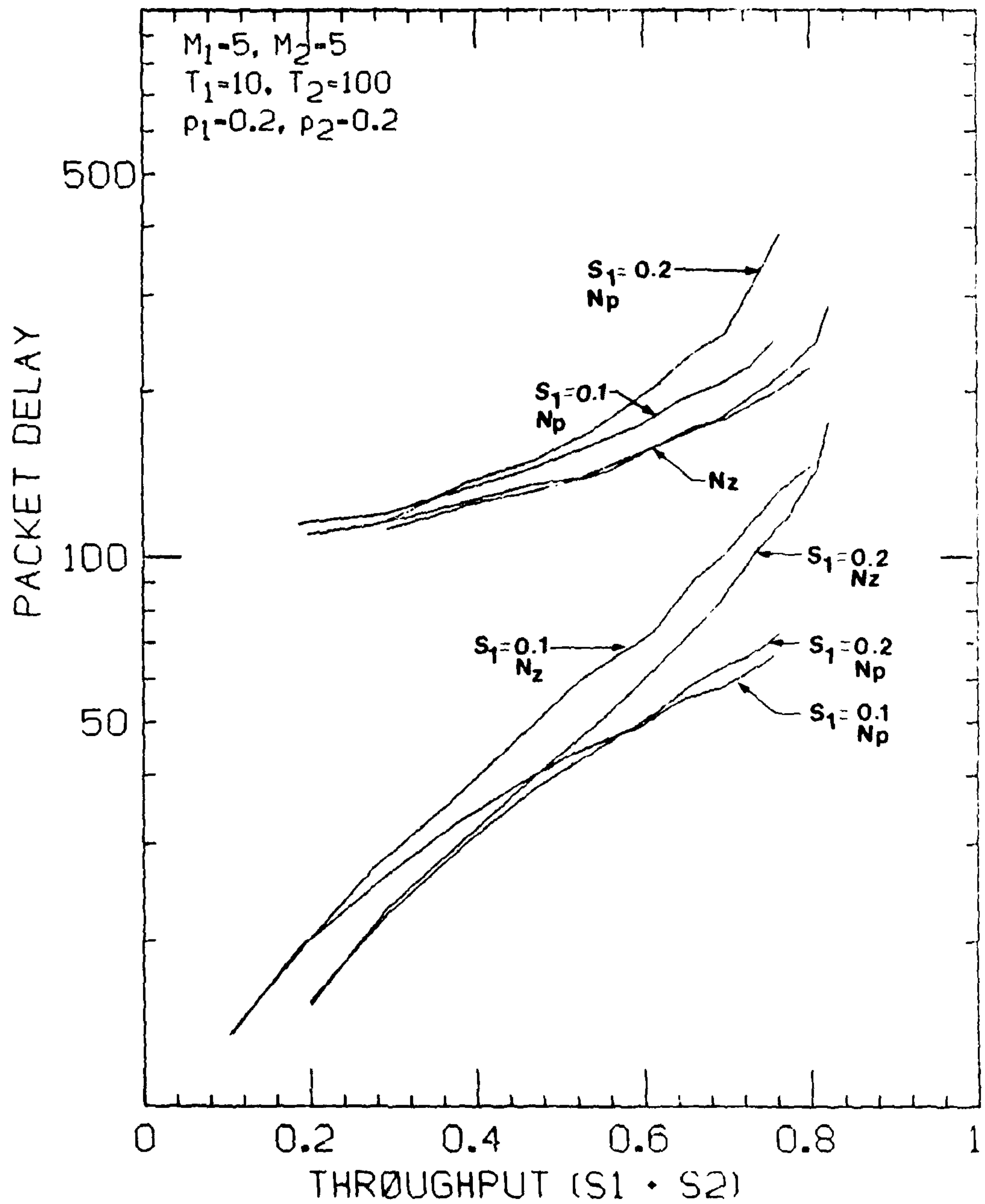


Figure 3. Throughput delay characteristics for nonpreemptive CSMA and nonprioritized CSMA with $S_1 = 0.1$ and $S_1 = 0.2$ respectively

different values of S_1 in the prioritized system. In the following we give an explanation for this difference and its behavior.

When $S = S_1 + S_2$ is small, the larger value of S_1 (0.2) gives a smaller volume of traffic for C_2 ($S_2 = S - S_1$) and allows C_1 packets to keep accessing the channel over a larger *number of subsequent contention periods* than with $S_1 = 0.1$. This results in a smaller packet delay for $S_1 = 0.2$ than for $S_1 = 0.1$ since for the same S, there are fewer C_2-packets which may cause long deference delays in this nonpreemptive mode. As $S_1 + S_2$ increases, this difference decreases until S becomes large enough so that S_2 is the predominant traffic in the channel, and a great fraction of C_1 packet arrivals takes place during the transmission periods of class C_2. In this situation, with the larger value of S_1 a larger number of C_1-packets accumulate at the end of a C_2 transmission needing to contend on the channel. With the same value of p_1 used for both values of S_1, $S_1 = 0.2$ provides higher delays than $S_1 = 0.1$.

The channel capacity is somewhat higher for the case of $S_1 = 0.1$ because of the effect that packet length has on channel capacity [8]: the larger the number of large packets transmitted in a network (which occurs when $S_1 = 0.1$) the higher the capacity will be as less time is spent in overhead (priority assessment, channel access, collision). If we compare the prioritized system to the nonprioritized system we see that in terms of delay for C_1 we have reduced the average delay significantly, especially when S is large, at the expense of larger delay for low priority packets. This is a desirable effect since we want high priority messages to have a finite delay and we don't restrict the delay of C_2.

In a semipreemptive system, C_1 packets can preempt C_2 packets when these low priority messages are in the channel access period (idle time of C_2 contention period). If we compare the nonpreemptive and semipreemptive mode, in small systems, we expect smaller delays for the semipreemptive case due to the preemption allowed, at the expense of larger delays of C_2-messages. In Figure 4 we show this comparison for a *small system* ($M = 5$). In this case semipreemption does not improve significantly the delay characteristics of the system for C_1; this is so because the channel access period is not very large with $p_i = 0.2, i = 1, 2$. This is not the case for large systems. We note that the improvement

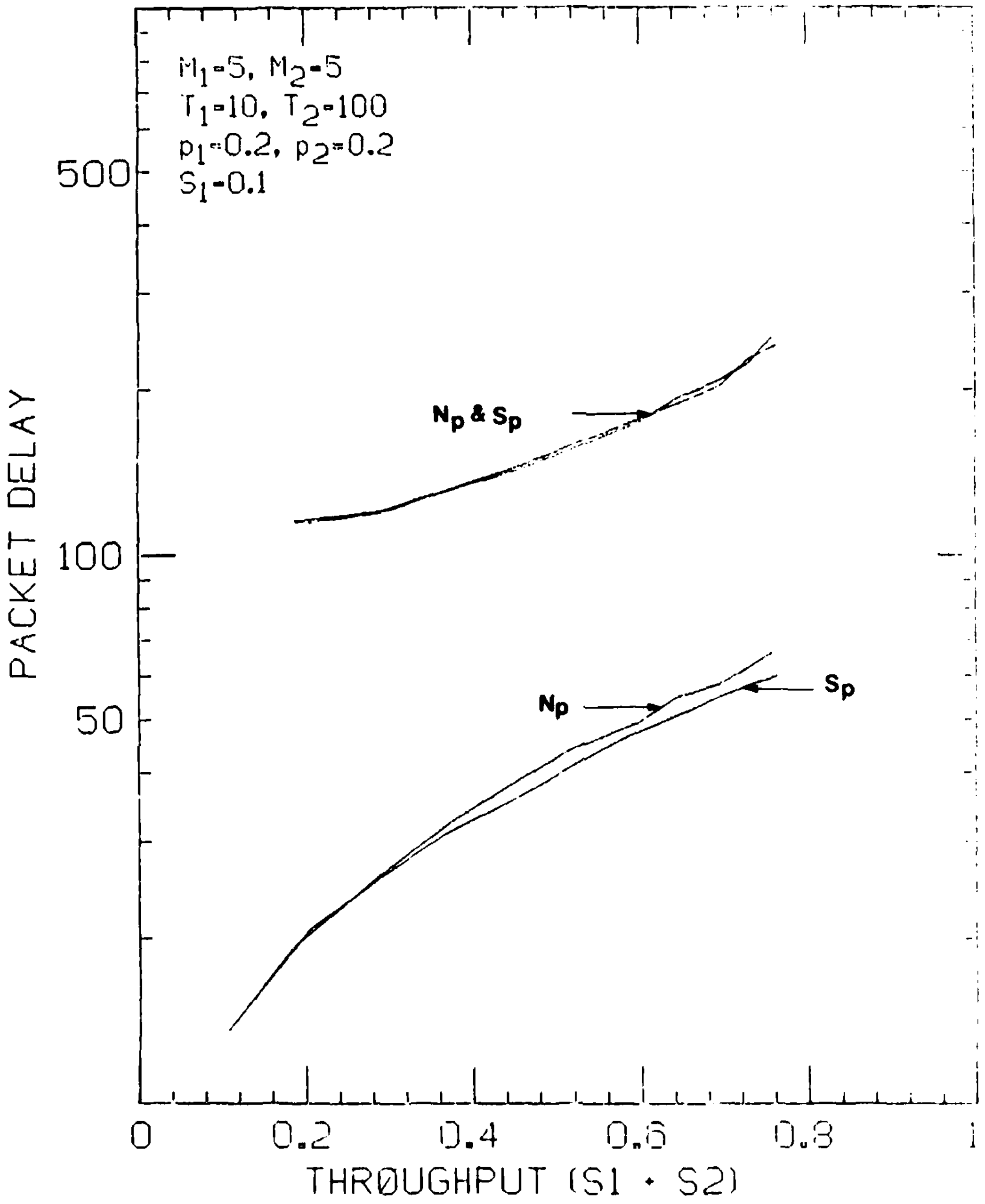


Figure 4. Comparison of average packet delay in nonpreemptive and semipreemptive systems

obtained in large systems is more important as will be shown later.

We now explore the effect of different levels of preemption on the throughput-delay characteristics. In Figure 5, we show, for different fractions (T_p) of C_2-messages in which preemption is allowed, how the packet delay for C_1 is reduced. This improvement is not without its cost however.Indeed with increasing T_p we can see that the delay for C_2 increases and the channel capacity decreases significantly. Note that $T_p = 0$ clearly corresponds to a semipreemptive system.

With full *preemption* ($T_p = 100$) we notice a decrease of delay for C_1 as the throughput aproaches channel capacity. Normally one would expect that, if full preemption is in effect, the delay will remain fairly constant as C_1 packets need only *cause a collision* when the channel is occupied by a low priority message. However, we notice an interesting variation of delay as the load increases (an increase followed by a slight decrease). This is explained by the following: when the load S_2 is low, most C_1 packets arrive when the channel is idle and undergo immediate first transmission, therefore incurring very small delays. As S_2 is increased and more C_2 packets arrive, more C_1 packets encounter transmissions from C_2 and are forced to cause a collision to start a new PAP, thus incurring higher delays. As the load of C_2 is further increased, to reach saturation, more C_1 messages will encounter collisions in C_2-transmissions. Assuming that C_1 messages that arrive to the system can detect ongoing collisions, C_1 packets wait for the PAP to begin, not having to incur the delay of causing the collision themselves. More importantly, they may arrive during the channel access period of C_2 preempting the class and succeeding sometimes in their first transmission, contributing to a smaller average delay. Overall, we have also achieved a reduction of the sensitivity of C_1-delay to the load exercised in the channel by C_2 which is a highly desirable effect.

Another effect of preemption is a reduction of the total channel capacity, caused by the thrashing that occurs in the channel as more time is wasted when ongoing transmissions are preempted. In Figure 6 we show this reduction in channel capacity as S_1 is kept constant at $S_1 = 0.1$, and the preemption fraction is varied. Packet delay for C_1 messages is greatly reduced (and delay for C_2 increased) as we increase the preemption fraction as

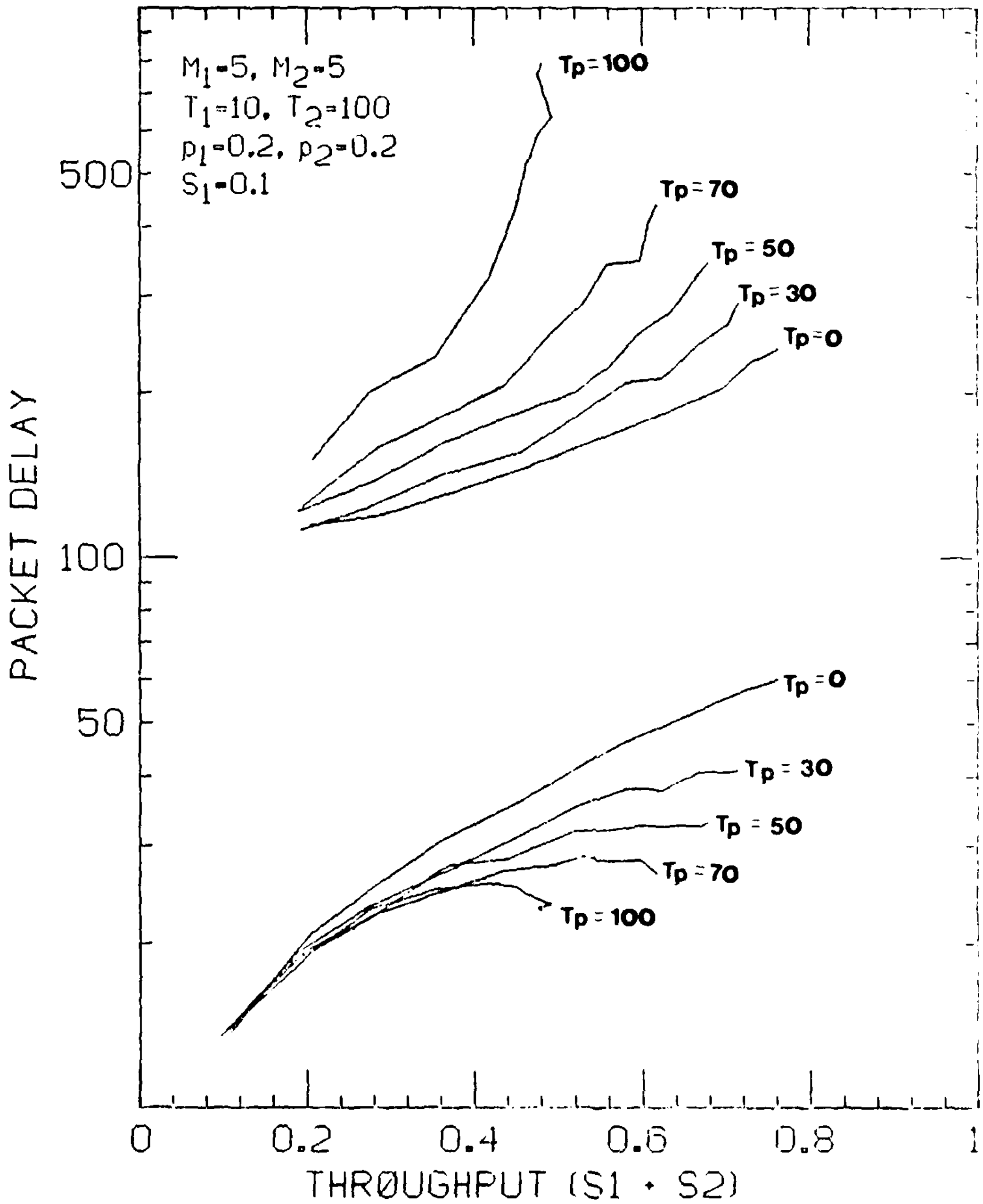


Figure 5. Throughput delay characteristics of a system for different preemption fractions (0,0.3,0.5,0.7,1.0)

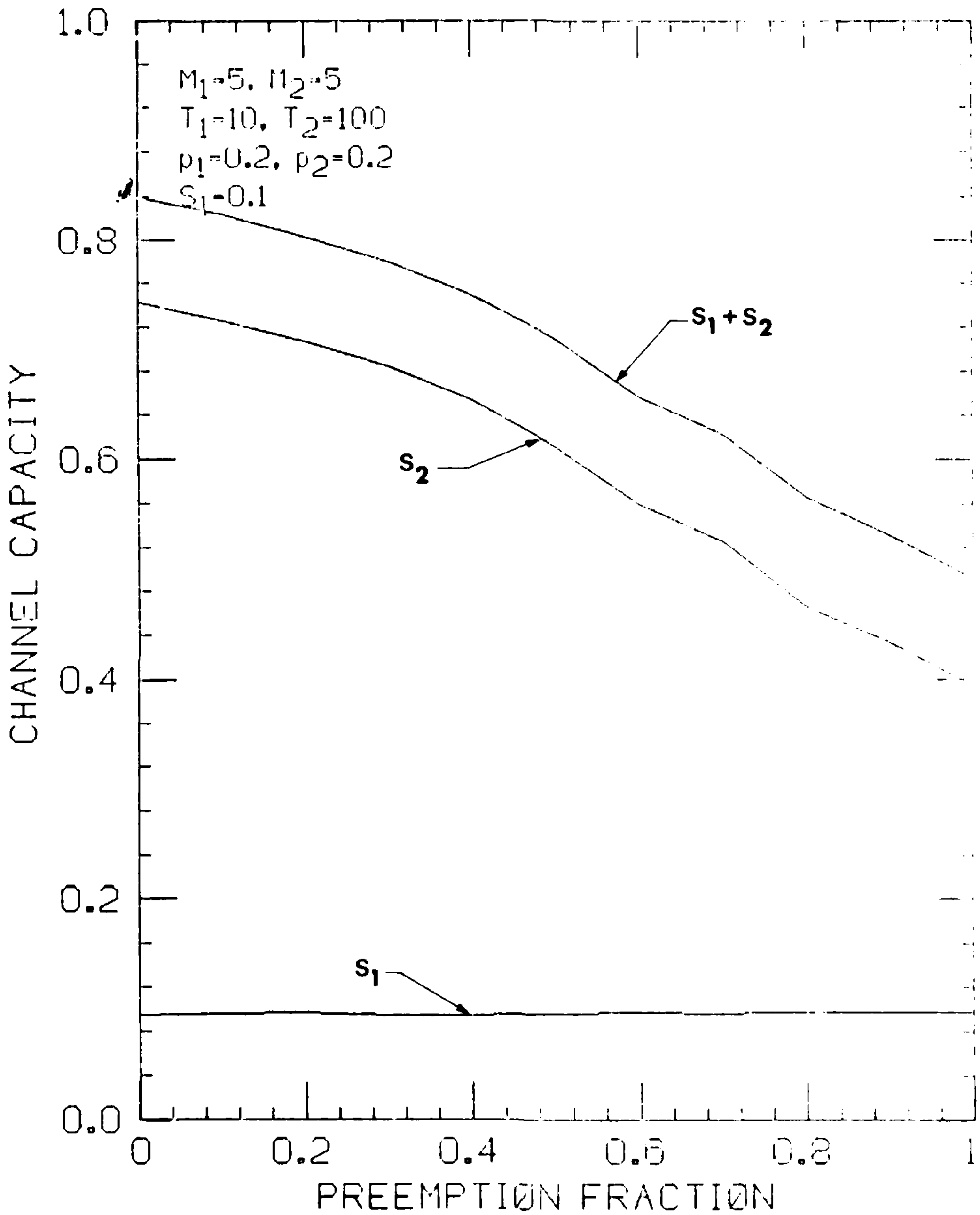


Figure 6. Channel capacity in a preemptive system as a function of the preemtion fraction

can be seen in Figure 7, where we plot the average delay per packet vs. the preemption fraction. Note the improvement achieved when full preemption is in effect ($T_p = 100$) as compared to semipreemptive systems ($T_p = 0$). Finally note that these overall effects caused by preemption, namely: a reduction of the delay for C_1 messages, increased delay for C_2 messages and a reduction of channel capacity constitute a tradeoff between C_1 delays on the one hand and total throughput and C_2 delays on the other hand. But as we have mentioned earlier, the need for priority functions arises when we have a channel in which we desire to multiplex an application with very tight delay constraints with another one that can tolerate large delays. In this case we are more interested in meeting the delay constraints than maximizing channel capacity or minimizing C_2 delay as long as traffic can be supported by the system. Our results are consistent with these objectives.

To summarize the results discussed so far we present in Figure 8 a comparison of all three modes of operation of P-CSMA-CD, and also included are the characteristics of a nonprioritized (obviously nonpreemptive) system.

In Figures 9-11 we present the throughput-delay characteristics of systems with $M = 50$ stations, packet lengths of $T_1 = 10$, $T_2 = 100$ slots, collision recovery time of $T_c = 2$ slots, $p_1 = 0.1$ and different values of p_2. In these systems we observe the same trends as in the small systems (when $M = 5$). The difference between nonpreemptive and semipreemptive, however, is more significant in this case where the choice of near optimum p_i is more critical. Note how this difference decreases from Figure 9 to Figure 11 as the choice of p_i gets better. It is very interesting to note the small sensitivity of C_1 delay to the choice of p_2 in the case of a semipreemptive system.We can explain this small sensitivity by the following: if the choice of p_2 is such that the channel access period is small, the time between the end of a transmission and the next C_2-arrival will be large. If, on the other hand, the channel access period for C_2 is large, the time until the next C_2 arrival will be small. Both, the C_2-channel access period and the idle time before a C_2 arrival, are times when C_1-packets can be transmitted, and this total time is the same, regardless of the choice of p_2. For nonpreemptive systems the choice of near optimum p_i becomes more critical as C_1 packets must wait the full length of the contention period for C_2 once the

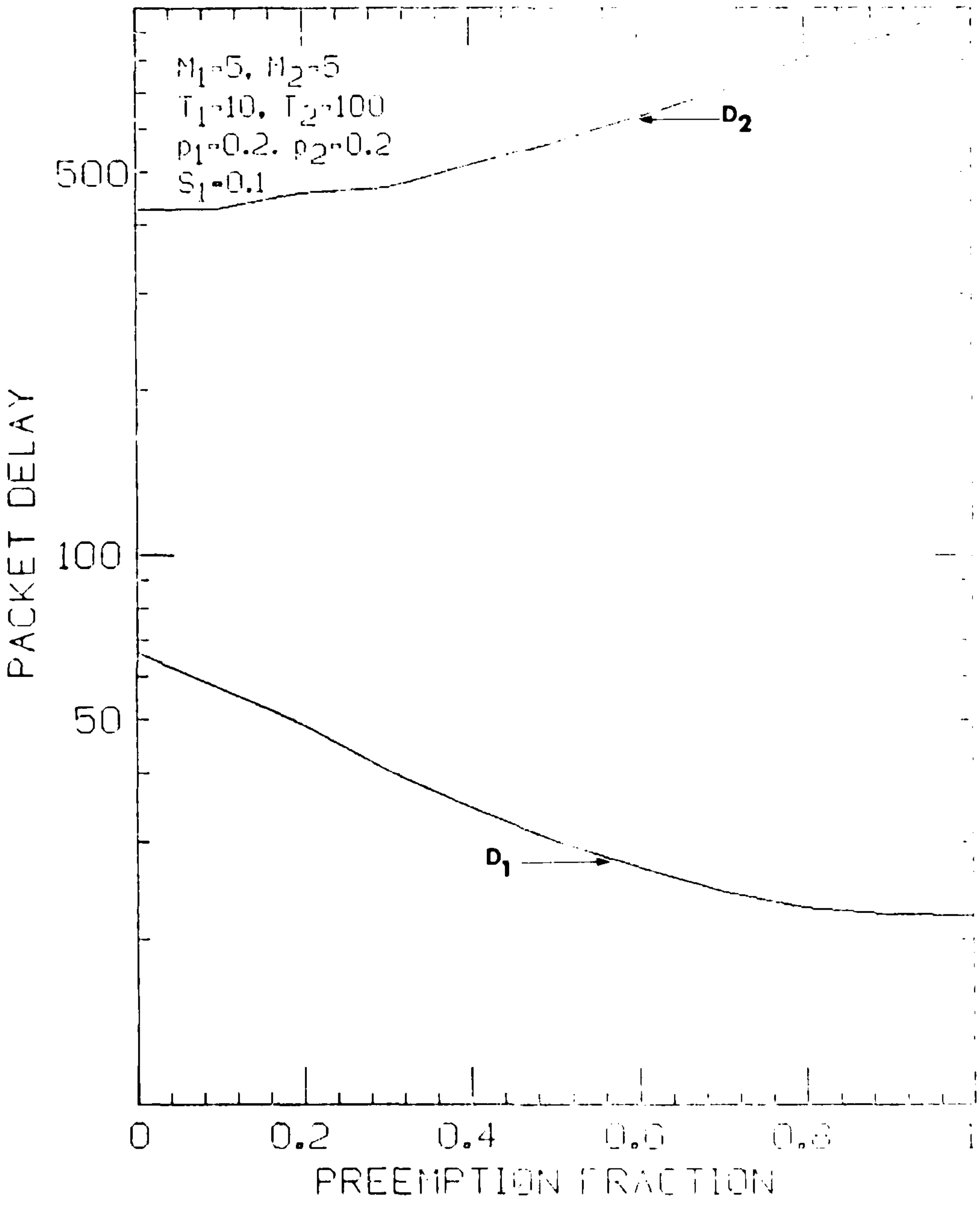


Figure 7. Average packet delay as a function of the preemption fraction

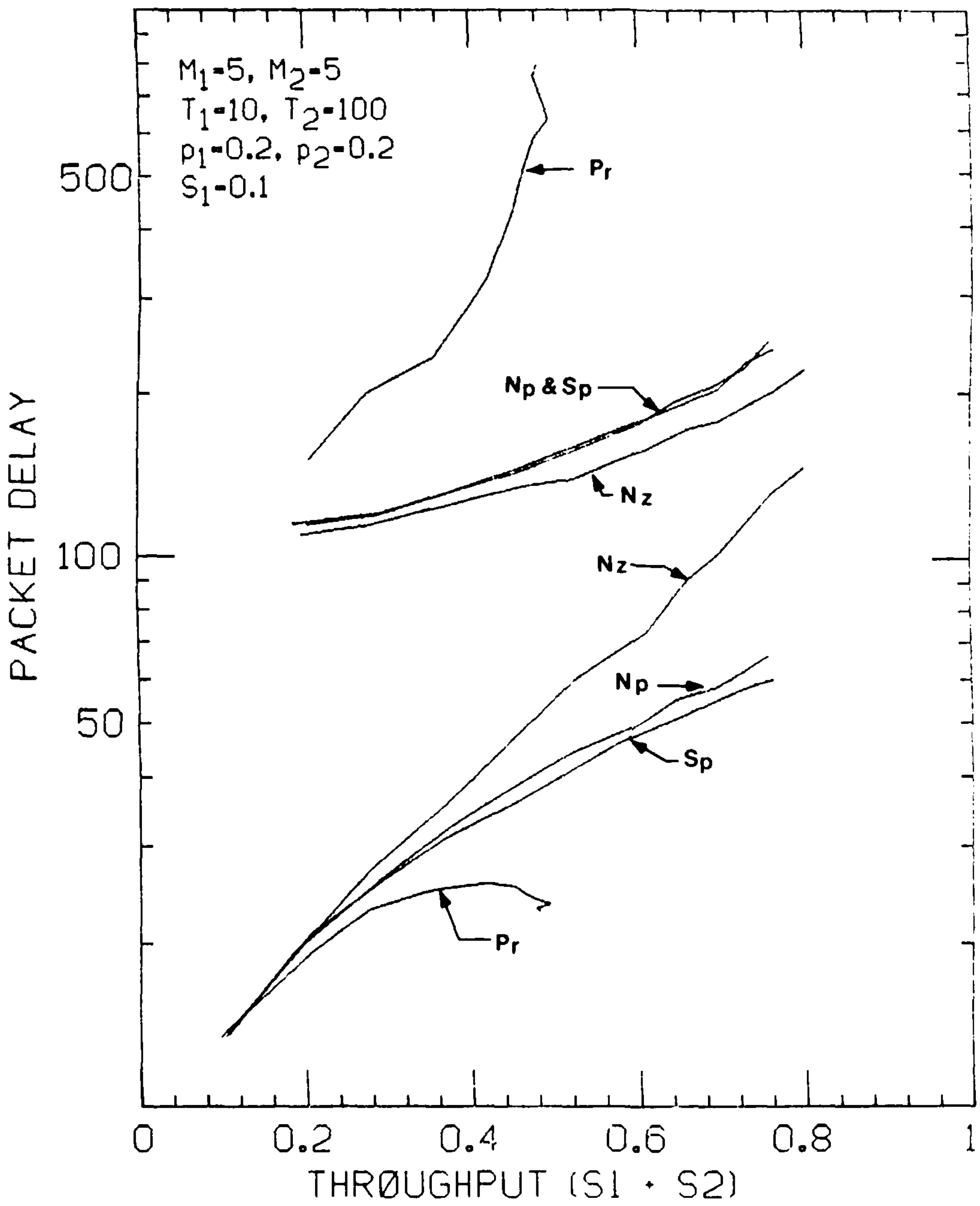


Figure 8. Comparison of average packet delay in prioritized and nonprioritized versions of CSMA

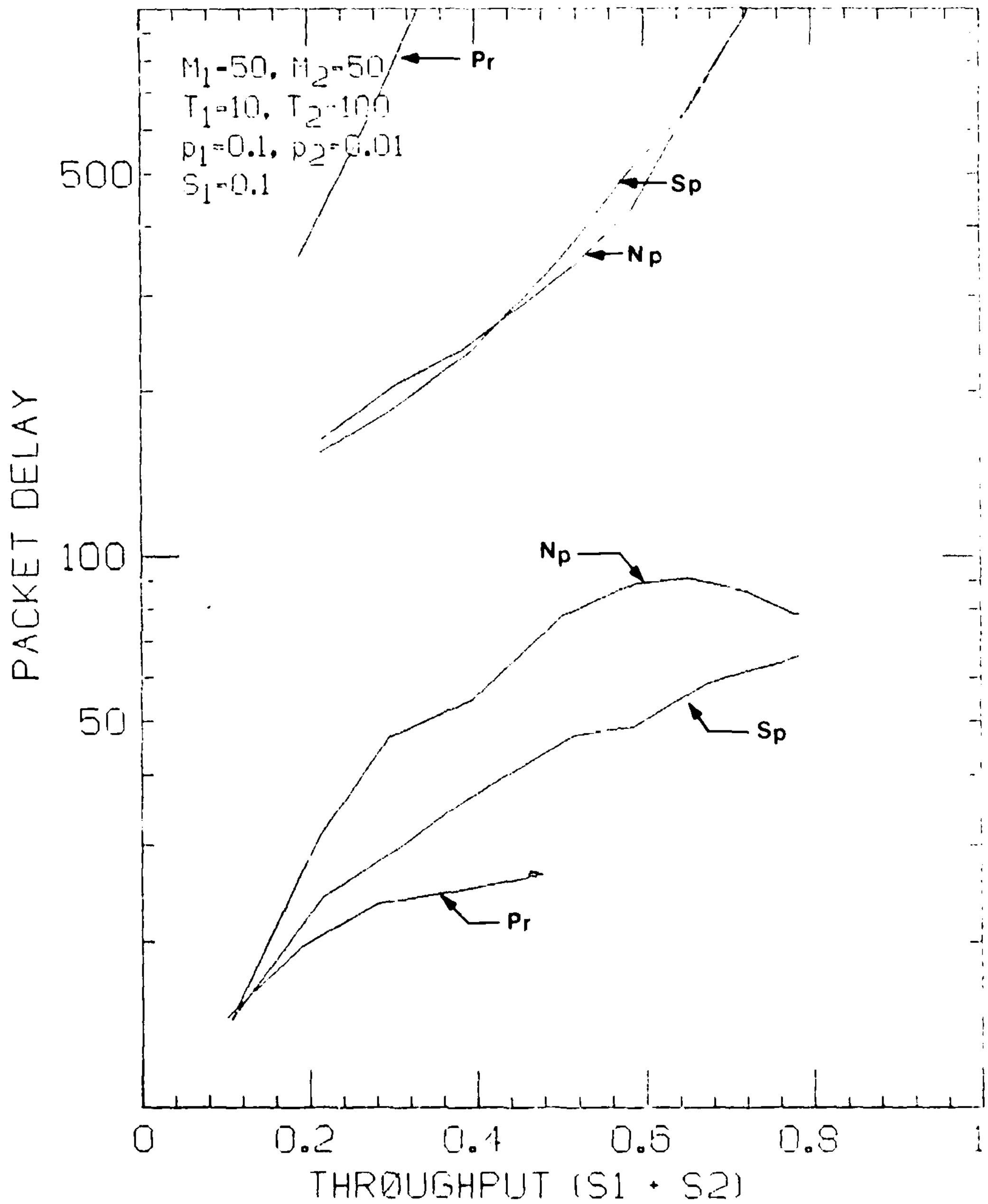


Figure 9 Comparison of average packet delay in the three modes of operation of prioritized CSMA for large (50 stations) systems with p = 0.01

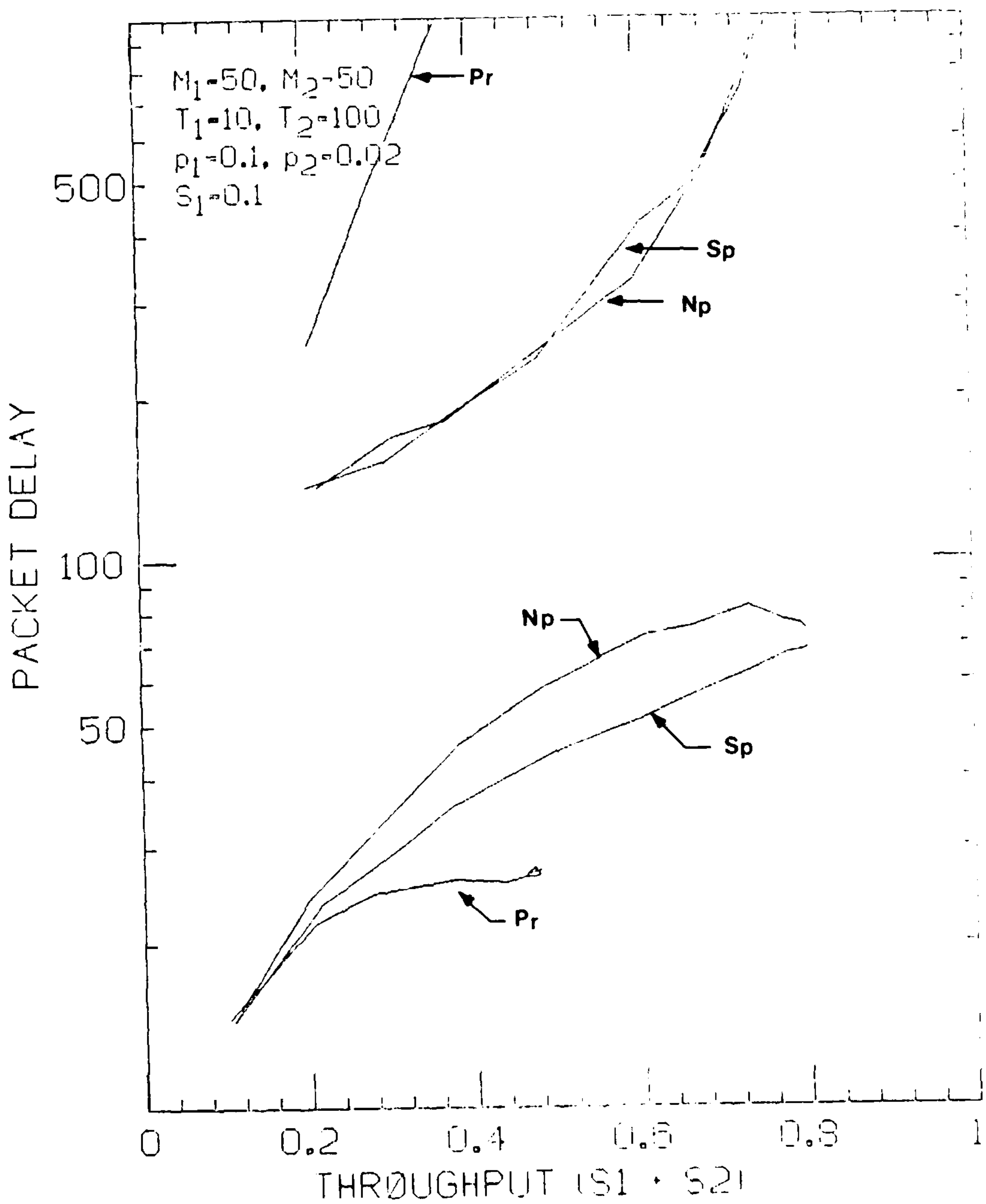


Figure 10. Comparison of average packet delay in the three modes of operation of prioritized CSMA for large (50 stations) systems with $p_2 = 0.02$

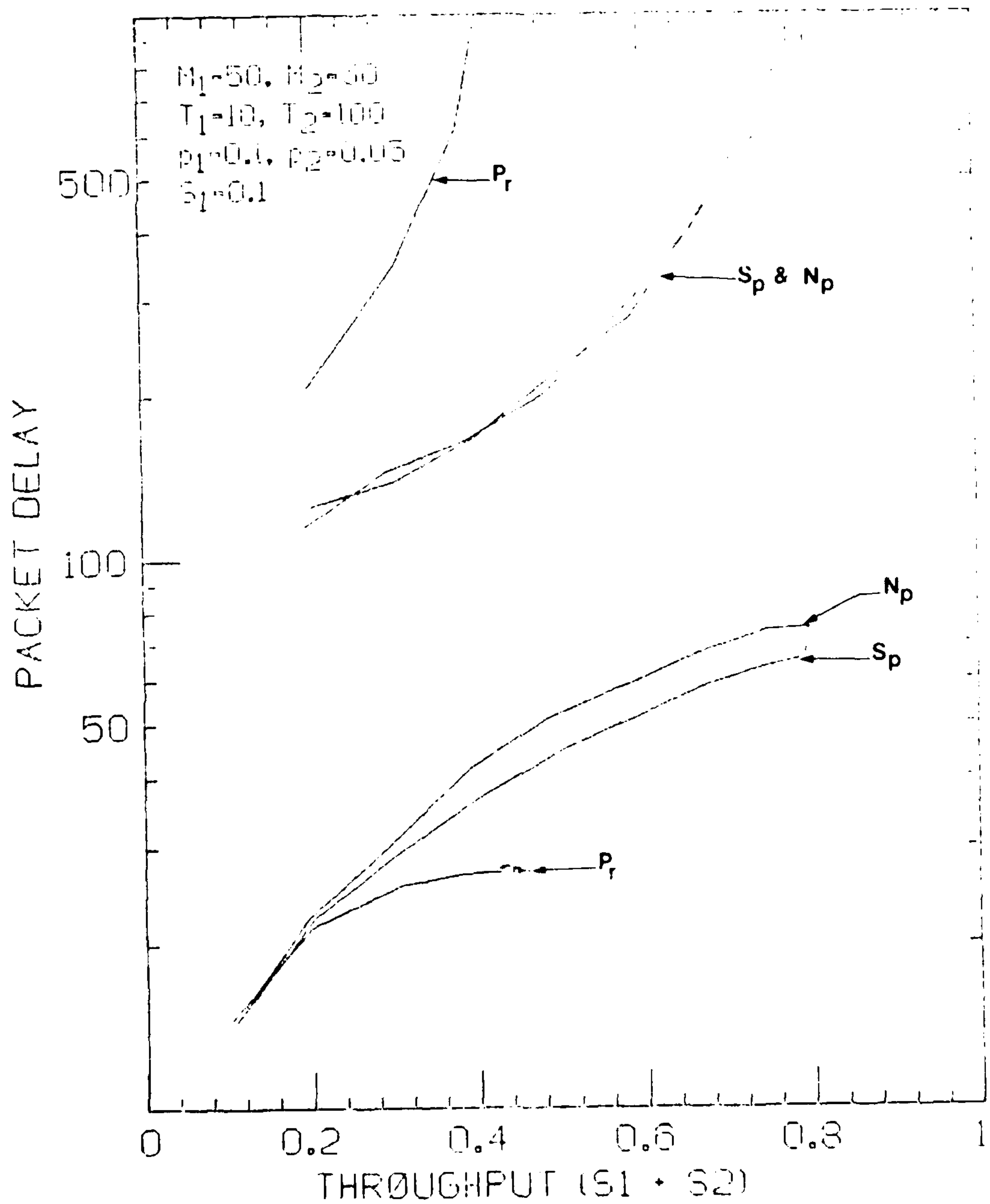


Figure 11 Comparison of average packet delay in the three modes of operation of prioritized CSMA for large (50 stations) systems with p = 0.05

channel has been granted to the low priority class. The difference between nonpreemptive and semipreemptive systems decreases as the load increases because the channel access period of C_2 becomes shorter at high loads. As *more stations try* to access the channel, either a transmission or a collision will occur sooner.

So far we have considered only the effect of preemption on system performance; another important aspect of performance that we now discuss is the effect of packet length. We examine these in small systems. In Figures 12-14 we examine the throughput delay characteristics of nonpreemptive and preemptive small systems with different packet lengths. In Figure 12 both classes transmit long messages ($T_1 = T_2 = 100$ slots). The difference between nonpreemptive and preemptive systems is still significant as the load gets high. The performance for C_2 is greately improved due to the small number of C_1 (long) packets *transmitted in* order to maintain the throughput S_1 constant ($S_1 = 0.1$). Note also that the channel capacity is not severely reduced by preemption. Again, the smaller number of C_1 packets, causes less thrashing in the channel as fewer C_2 packets get preempted by incoming C_1-messages. Figure 13 shows the characteristics of a system in which both classes transmit short packets($T_1 = T_2 = 10$ slots). In terms of packet delay the performance is similar to the case when all the packets are long. But note the reduction in channel capacity for this case due to the smaller packet length. Since a larger number of packets have to be transmitted in order to maintain a certain throughput, more *time* is spent in overhead such as reservation and channel access periods. Consequently the channel capacity decreases significantly.

In Figure 14 we consider the performance characteristics of a system in which high priority messages are long and low priority messages are short *($T_1 = 100, T_2 = 10$ slots)*. Notice the similarity between the performance of nonpreemptive and preemptive systems, both in packet delay and channel capacity. The similarity in packet delay is caused by the fact that preemption *does not* save a great deal of time, because an ongoing transmission that can be preempted is a short one. Even if a packet waits until an ongoing transmission is finished, as it is the case in the nonpreemptive system, such a packet would only have to wait a few extra slots before the next priority assessment period begins. By causing a

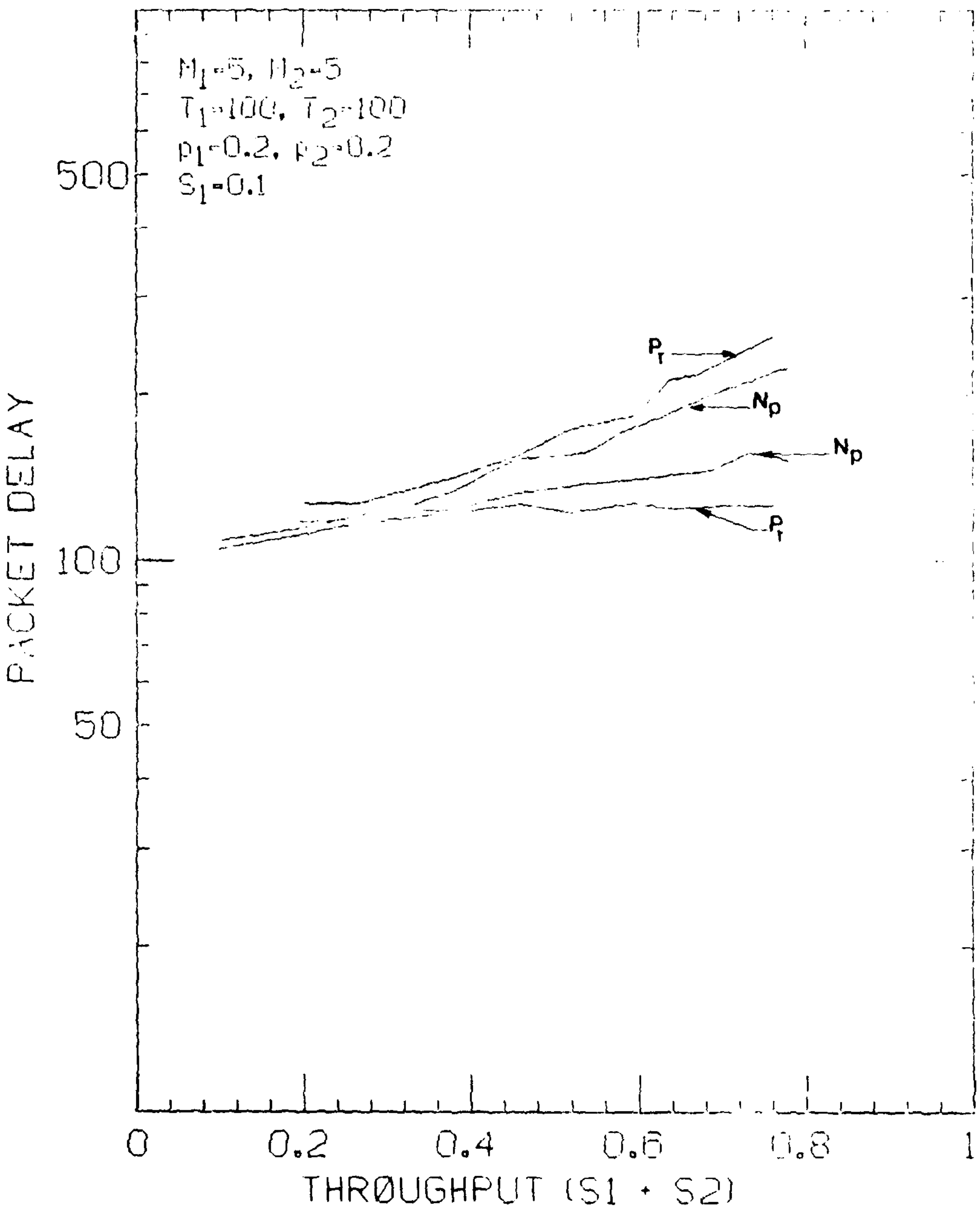


Figure 12. Comparison of average packet delay in nonpreemptive and preemptive systems with packet lengths of $T_1 = 100$ and $T_2 = 100$

collision and wasting the recovery time not too much time is saved, hence the similarity in delay for the two modes. The reduction in channel capacity is *due to the packet length* of C_2 ($T_2 = 10$ slots) [8]. The majority of the packets that are transmited are short packets. As we saw in Figure 13, this causes a reduction of the total channel capacity. The similarity between the performance of the nonpreemptive and the preemptive system in terms of channel capacity is due to the small *amount of thrashing* that occurs. In this case long packets, which are of high priority, cannot be preempted. If preemption does occur, the preempted packet is a short one, causing a relatively small amount of thrashing. Furthermore, since S_1 is low, a very small number of C_1-packets are transmitted in the channel. In the long period of time between two consecutive C_1 arrivals a large number of backlogged C_2-packets can be successfully transmited. The system, then, behaves very smilar to a nonpreemptive system simply because preemption is not exercised very often.

Preemption, therefore, is valuable only when the time we save by preempting an ongoing transmission is significantly greater than the time wasted by causing a collision. What is considered significantly greater will, of course, depend on the application constraints.

Another aspect considered here, which was not included in the analysis[12], is the effect that buffer size has on the overall performance of the system.

We considered in this work systems with one and two buffers per station only. The results obtained clearly show the effects of this feature.

In Figure 15 we present the results obtained for a nonpreemptive system with $M = 5$ stations, $p_1 = p_2 = 0.2, T_1 = 10, T_2 = 100$ slots and $T_c = 2$ slots. Two cases are presented: $B_i = 1$ and $B_i = 2$ buffers per station, $i = 1, 2$. The delay for C_1 packets is not significantly different in the two cases because C_1-packets do not *incur* very large delays and very seldom will a station have its two C_1-buffers full. In the case of C_2 however, the difference in delay is much more significant as many of the packets that are lost when $B_2 = 1$ are now accepted into the system to occupy the second C_2-buffer. These packets incur large delays before they are transmitted whereas in single buffer systems they were

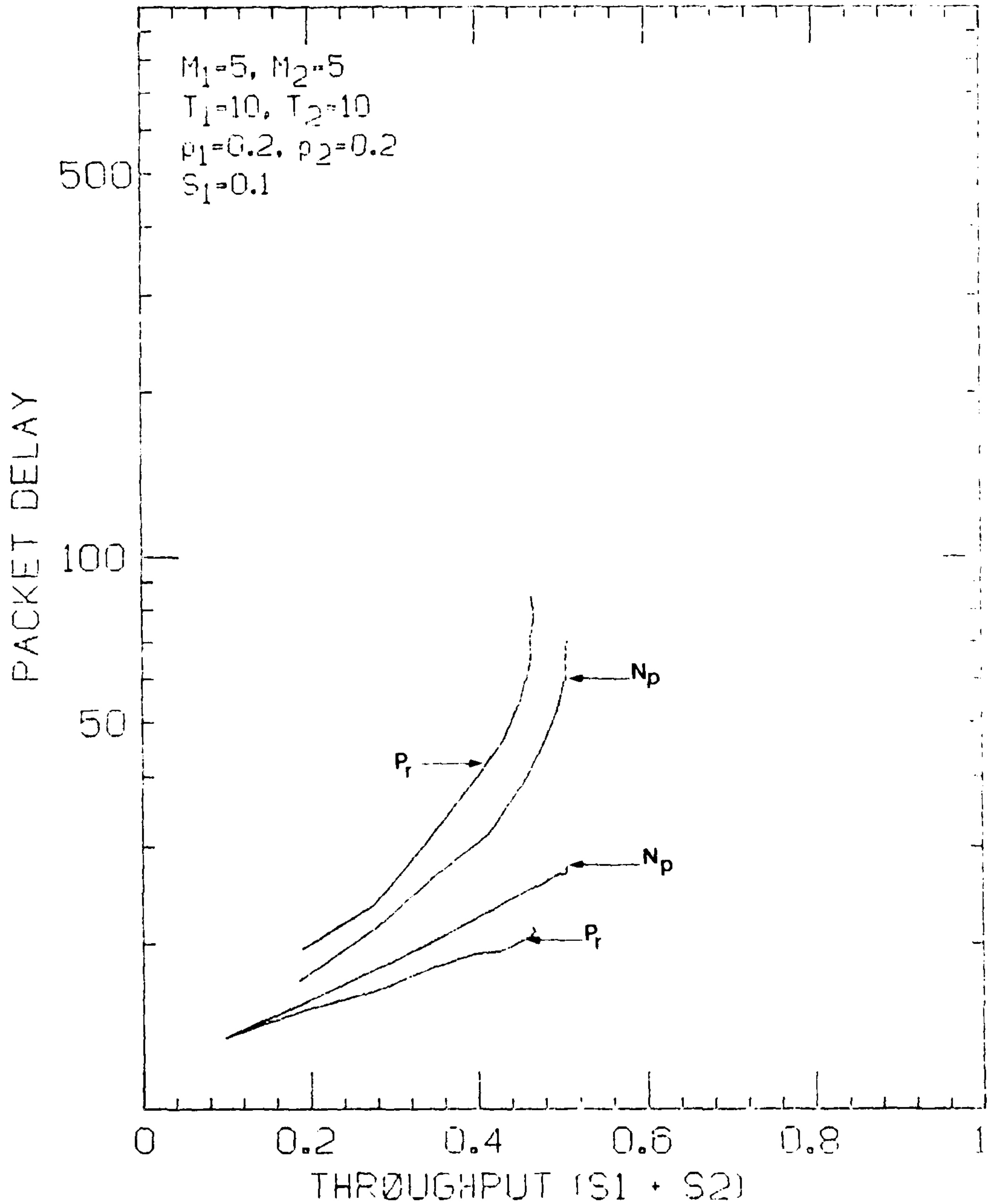


Figure 18. Comparison of average packet delay in nonpreemptive and preemptive systems with packet lengths of $T_1 = 10$ and $T_2 = 10$

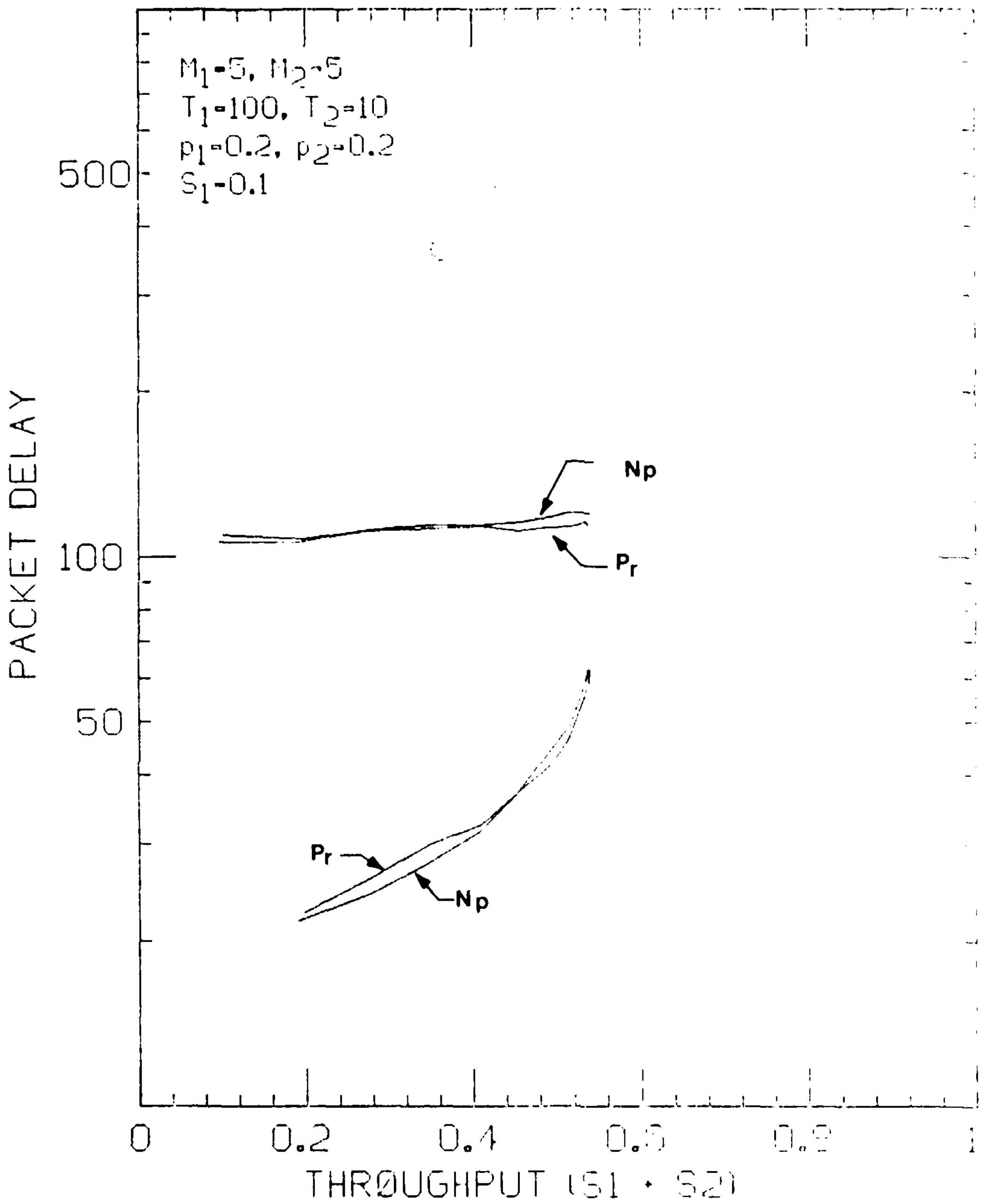


Figure 14. Comparison of average packet delay in nonpreemptive and preemptive systems with packet lengths of T_1 = 100 and T_2 = 10

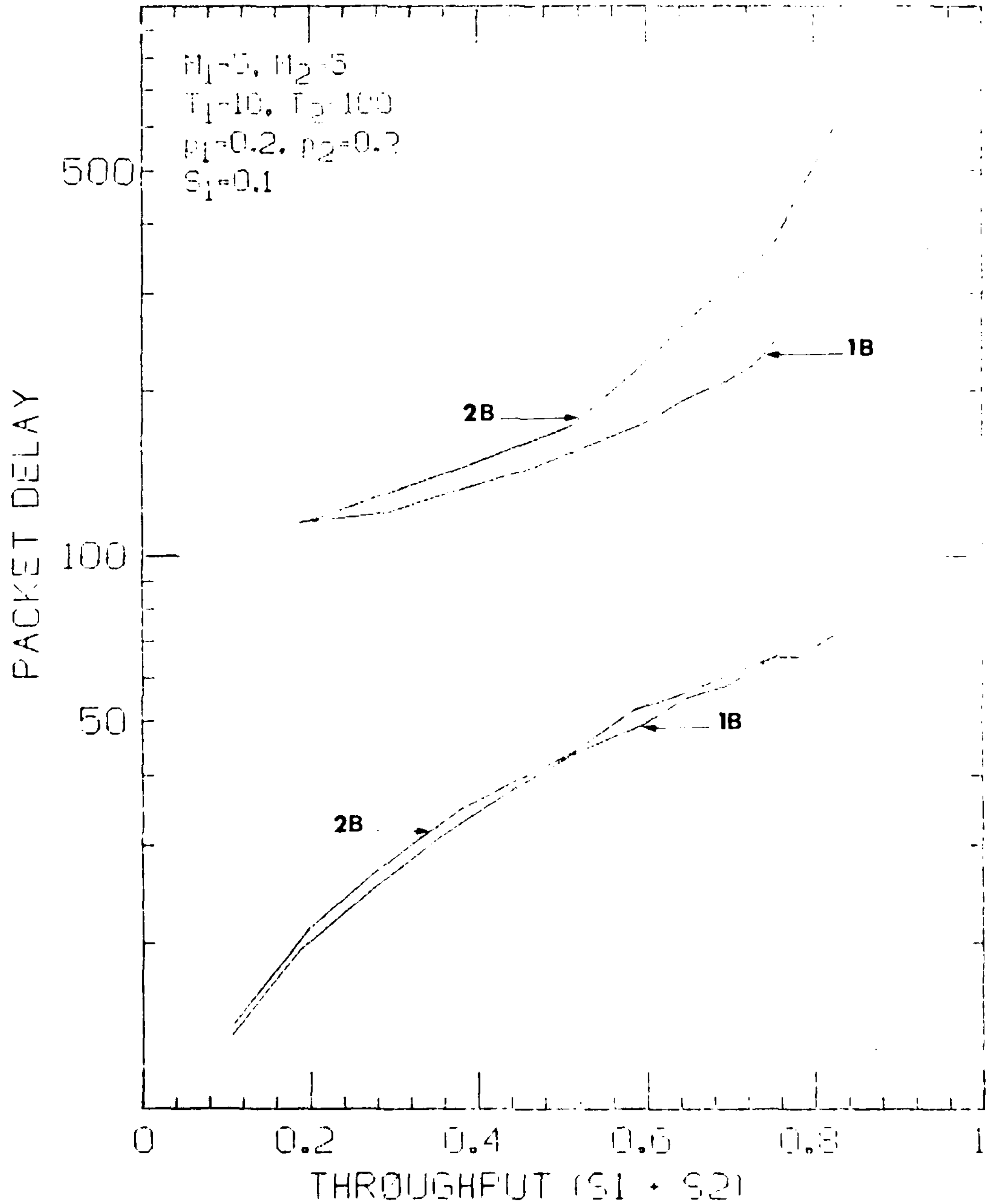


Figure 15. Average packet delay in nonpreemptive systems with 1 and 2 buffers per station

simply lost,not contributing to the average packet delay.

Figure 16 shows the characteristics of a preemptive system with the same configuration specified above. Note here that the difference in delay for C_1 in the two cases shown is even smaller since the second C_1-buffer is practically never used by the stations. Again C_2-packets in two buffer systems experience larger delays for the same reasons than in nonpreemptive systems.

We compare nonpreemptive CSMA-CD and nonprioritized CSMA-CD for the two cases: $B_i = 1$ and $B_i = 2$, $i = 1,2$, in Figure 17. In nonprioritized systems the delay for short packets grows very rapidly. The difference in this delay between the one and two buffer systems shown is also significant. This is in contrast to the delay achieved with the *introduction* of the *nonpreemptive* priority function, which is similar in the two cases shown. The advantage of adding an extra buffer in the system is in terms of packet loss, which is discussed in the next subsection.

The final aspect of system performance that we consider is the effect of collision detection. We *cannot make* the scheme dependent on collision detection if we want to use it in *radio environments* where this feature is not available.

We show a comparison of the performance of systems with and without collision detection in Figure 18. The systems shown without collision detection are nonpreemptive and semipreemptive; a preemptive system was not considered as it is not implementable in systems without collision detection. The systems considered have $M = 50$ stations, $T_1 = 10, T_2 = 100$ slots, $p_1 = 0.1, p_2 = 0.2$ and where applicable $T_c = 2$. It may seem surprising that the most significant difference in system performance is in channel capacity and not in packet delay. We must remember that the only oportunities for collision in a *nonpreemptive* system are during the channel access period, when two packets of the same priority class transmit at the same time. With a good selection of p_i the probability of such a collision is greatly reduced. In the semipreemptive system another opportunity for a collision is during the channel access period of low priority classes. Again a good selection of p_i coupled with the fact that the probability of a packet arrival during such a short

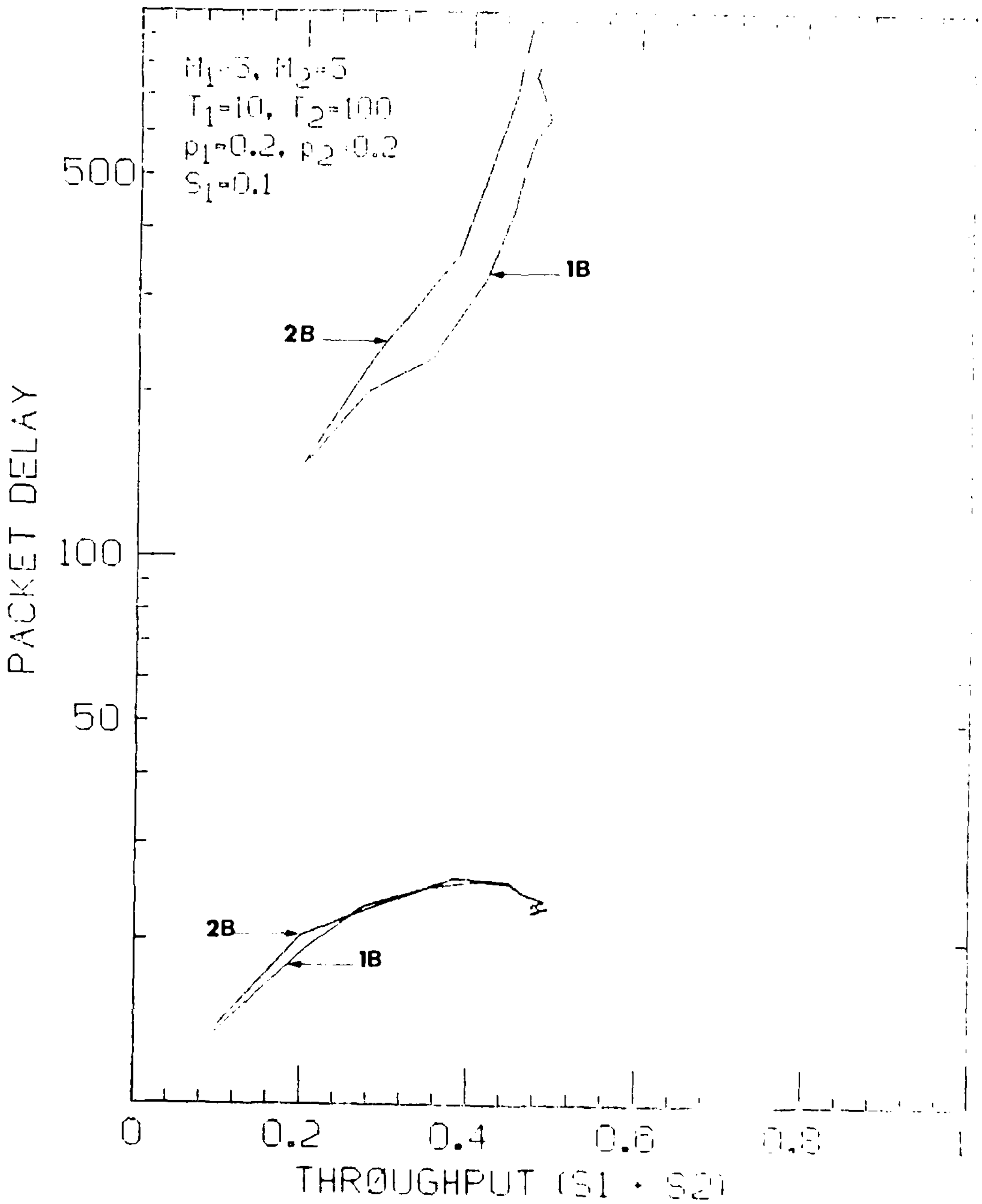


Figure 16. Average packet delay in preemptive systems with 1 and 2 buffers per station

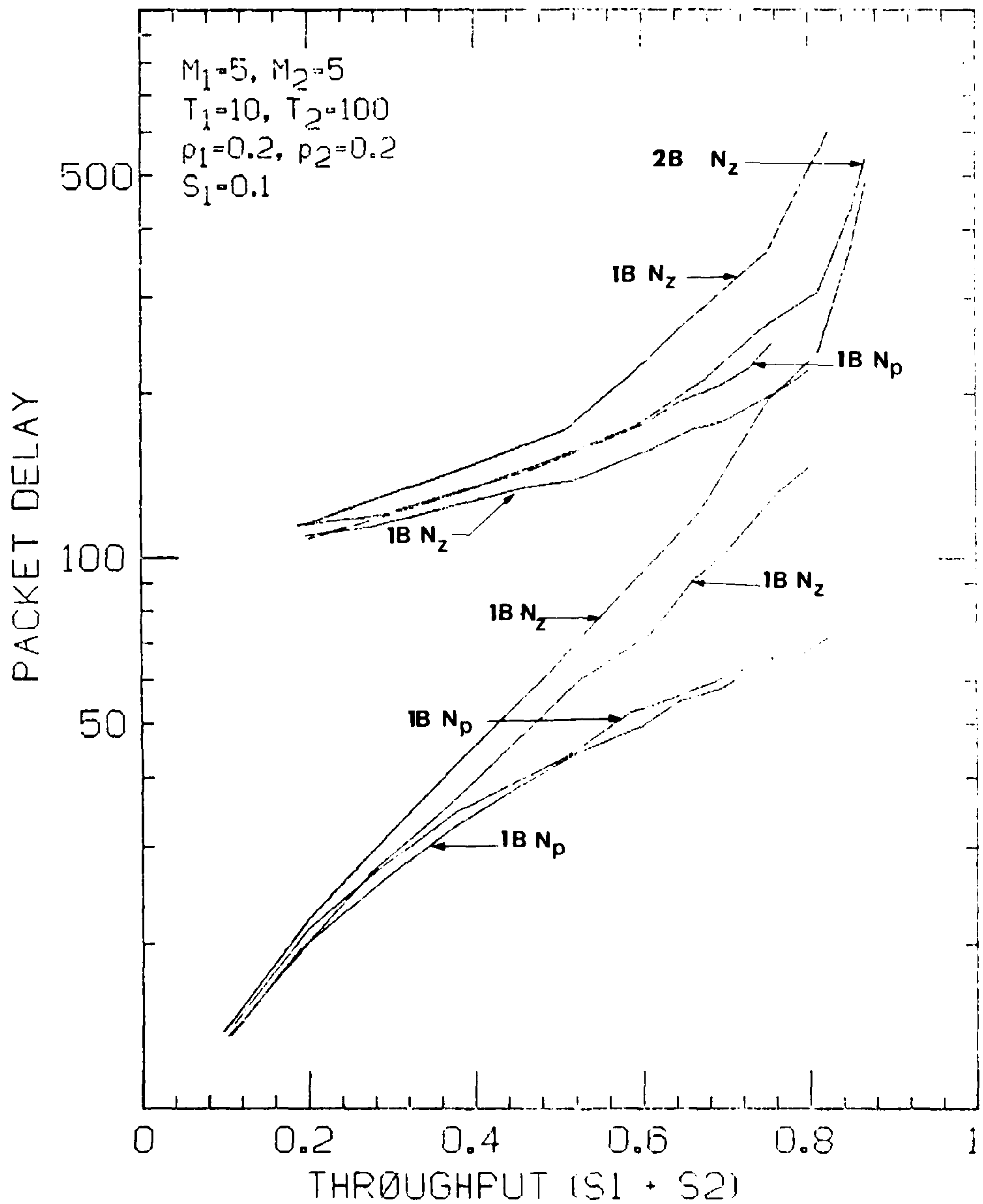


Figure 17. Comparison of prioritized and nonprioritized systems with 1 and 2 buffers per station

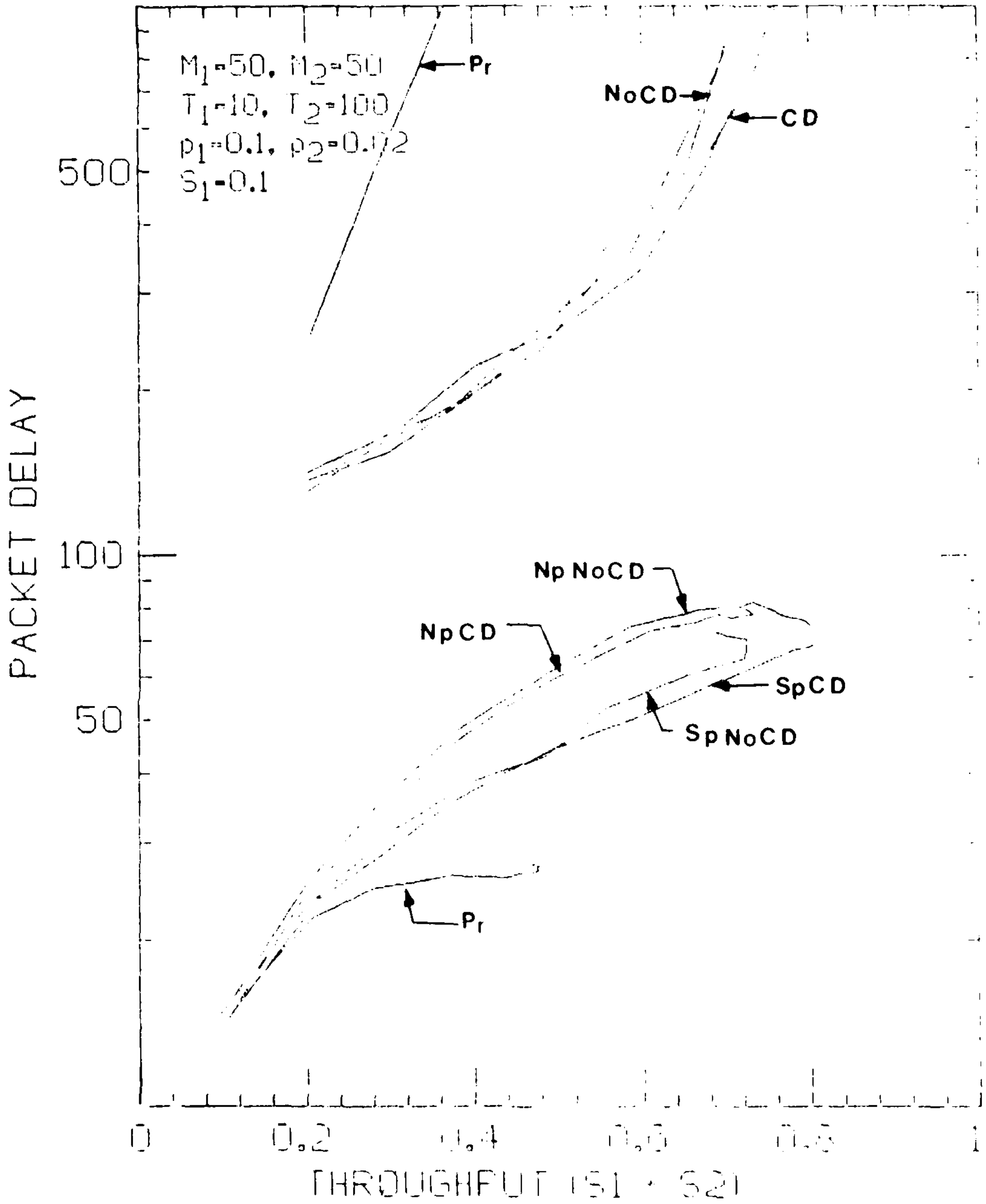


Figure 18. Comparison of average packet delay in large (50 stations) systems with and without collision detection (B = 1)

period is very small reduce the likelihood of a collision. It is the fact that not too many collisions occur which reduces the difference in performance for systems with and without collision detection. This is an encouraging result for the application of the scheme in radio environments.

5.2 Packet Loss

Packet loss was considered a major aspect of system performance because of the importance it has in some applications that require high priority in multiaccess channels. Packet loss can be due to the limited number of buffers in the system. In some real time applications, such as voice transmission, the generation process is independent of network response and packet loss can also be due to packet delays greater than the maximum allowable. We study in this subsection the ratio of packets lost to the total number of packets generated. We begin our discussion by considering a nonpreemptive system with $M = 5$, $B_1 = B_2 = 1$, $T_1 = 10$, $T_2 = 100$ slots and different values of p_i, $p_1 = p_2 = 0.1, 0.2, 0.5$. We show the packet loss for this system in Figure 19. Note that, as in packet delay, the loss in this system of small size is not too sensitive to variations in the choice of p_i. The loss of C_1 is significantly below that of C_2 and, in the cases shown, never exceeds 16%. As the channel approaches saturation, the loss for C_2 approaches 100% as a consequence of the large delays incurred by C_2 packets in this range of high channel throughput S.

If we increase the number of stations to $M = 50$, packet loss is significantly reduced. We observe this effect in Figure 20. This reduction in packet loss is due to the higher number of buffers available in the system. In order to maintain a given aggregate load, stations must generate packets at a smaller rate than in the case $M = 5$ stations. The average interarrival time of packets at any given station becomes larger and the station can tolerate larger delays before losing packets. As the channel approaches saturation, the packet loss for C_2 still approaches 1. Clearly this will always be the case for any finite number of stations. The reduction in packet loss for C_2 over the throughput range shown

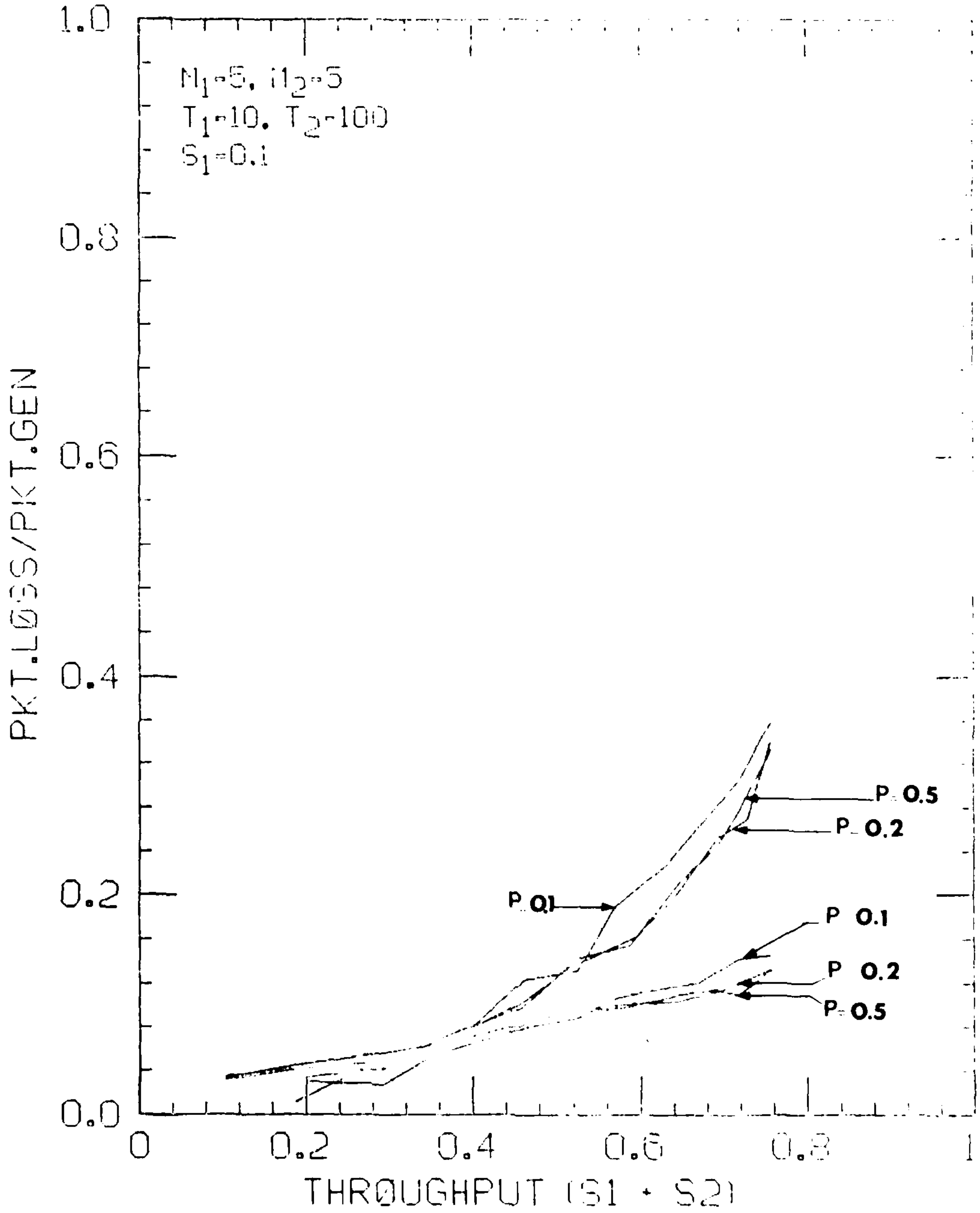


Figure 19. Packet loss in a small nonpreemptive system with different values of p (0.1,0.2,0.5) (B = 1)

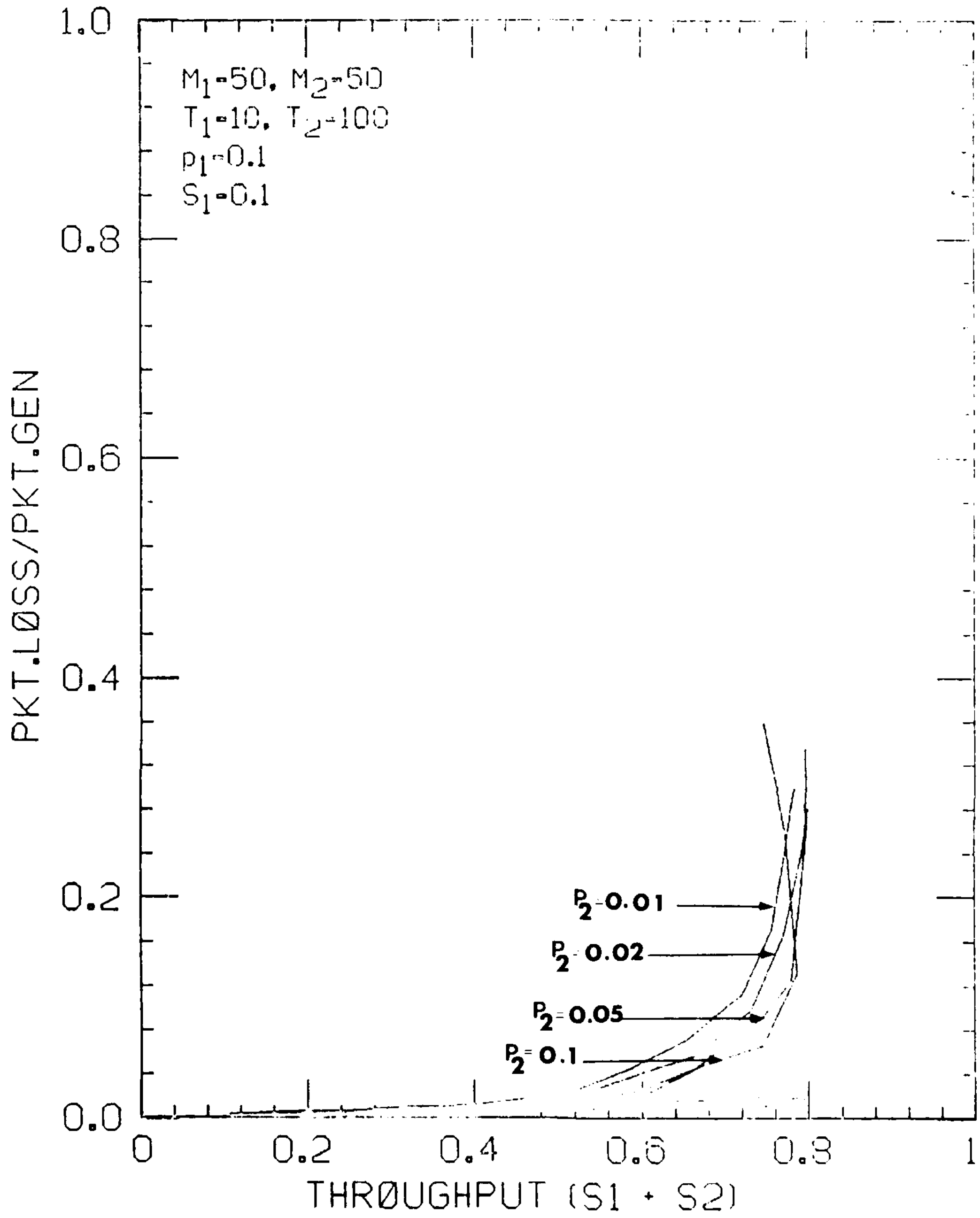


Figure 20. Packet loss in a large nonpreemptive system with different values of p_2 (0.01,0.02,0.05,0.1) ($B = 1$)

can be appreciated by the displacement of the 'knee' of the curve closer to the saturation point.

We have mentioned that packet loss and packet delay are related. If we experience smaller packet delay we expect a smaller packet loss. One can further reduce packet loss by adding buffers to the system. In Figure 21 we compare the packet loss characteristics of small systems ($M = 5$) with $B_i = 1$ and $B_i = 2$, $i = 1, 2$. As expected we see a significant reduction in the packet loss of both priority classes when we add one extra buffer per class at each station. For class C_1, packet loss becomes [illegible]. Indeed, due to the small delays incurred by C_1 packets [illegible] be used, as we mentioned in the previous subsection. [illegible] the loss of a C_1-packet, which occurs when it arrives to a [illegible]. For C_2, the addition of one extra buffer has displaced the knee of the curve to the right, closer to the saturation point. Once again, as we approach channel capacity, the loss for C_2 grows toward 1, which will happen for any finite number of buffers per station.

Comparing now P-CSMA to nonprioritized CSMA, we consider $M = 5$, $T_1 = 10$, $T_2 = 100$. We show the results in Figure 22. Note that, for $B_1 = B_2 = 1$, the loss for short and long packets in nonprioritized systems follows the same pattern and approaches 1 as the throughput gets closer to channel capacity, while in prioritized systems the slope of the curve for C_1-loss is smaller. Note the excellent improvement in the prioritized systems when the number of buffers per station is increased to 2 as compared to nonprioritized systems.

In the previous subsection we showed the reduction of delay for C_1 messages caused by preemption. Due to the close relationship between packet delay and packet loss, we also see a significant reduction of packet loss when preemption is allowed. This reduction is shown in Figure 23 where we plot the packet loss for one buffer systems in the nonpreemptive and fully preemptive modes. Note the small percentage of packets lost in the preemptive system over the entire range of throughput shown. In general packet loss will decrease as we increase the fraction of the transmission period in which we allow preemption, as we show in Figure 24. We can see that the packet loss is reduced by a

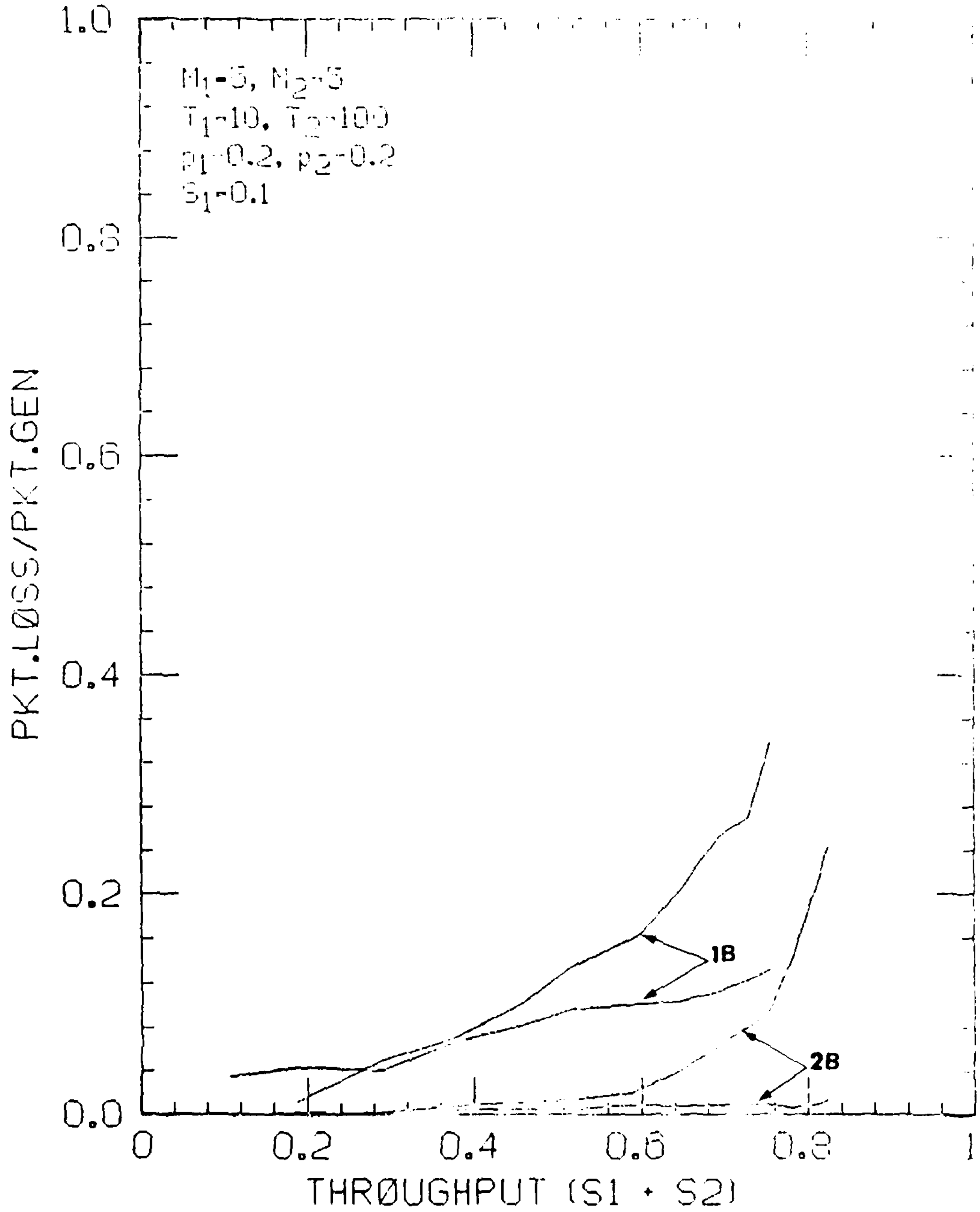


Figure 21 Packet loss in nonpreemptive systems with 1 and 2 buffers per station

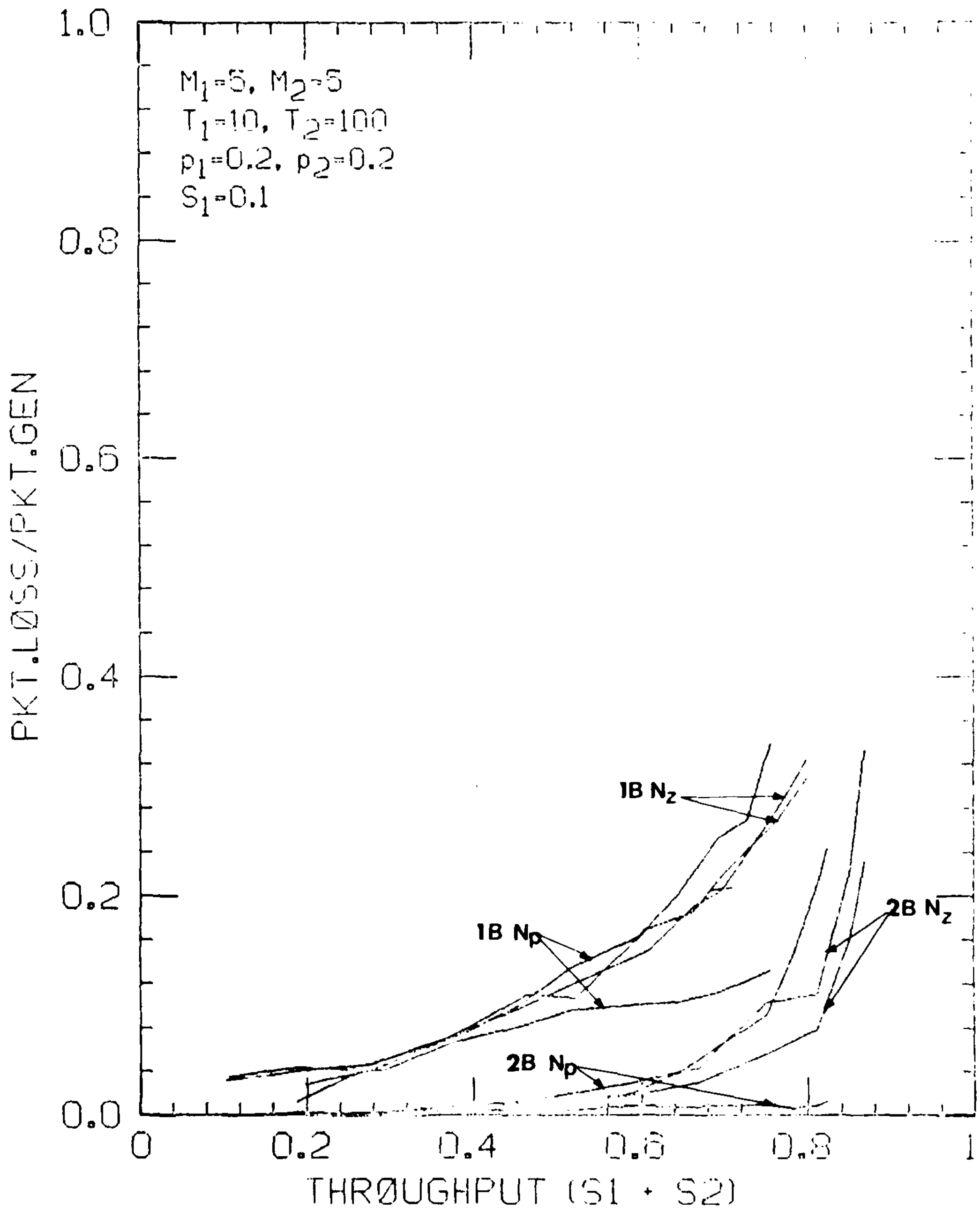


Figure 22. Comparison of packet loss in prioritized and nonprioritized systems with 1 and 2 buffers

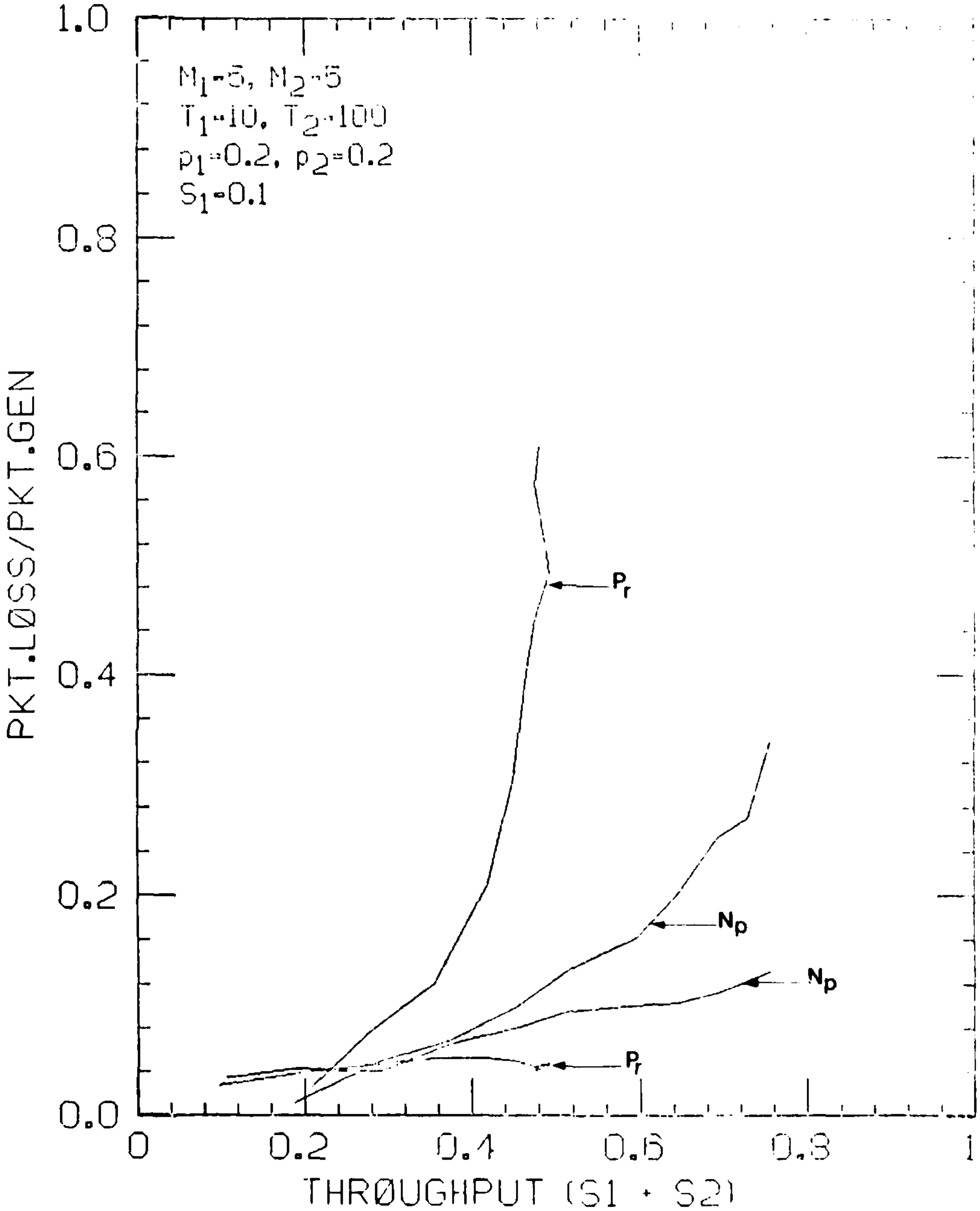


Figure 23. Comparison of packet loss in preemptive and nonpreemptive systems

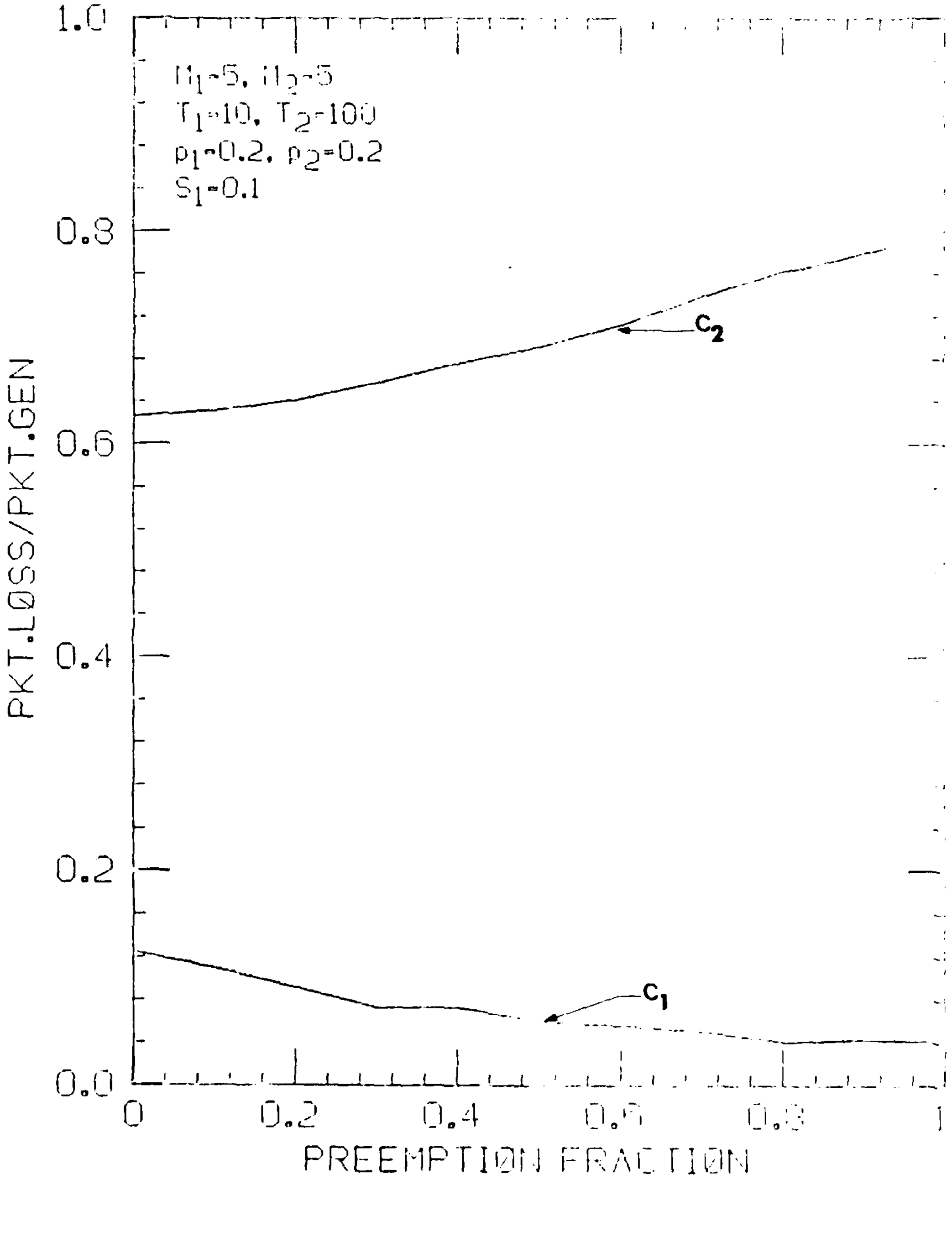


Figure 24. Effect of preemption fraction in the packet loss of the system

factor of 3 (from 16% to 5% from a semipreemptive to a full preemptive discipline). Again, a *remarkable improvement is obtained* by adding one extra buffer to each station. This improvement is shown in Figure 25, where we can see that, for preemptive systems with two buffers per station, C_1-packet loss is practically negligible over the entire range of S.

Once again, the improvement shown is not restricted to systems with collision detection. In Figure 26 we show the packet loss of a nonpreemptive P-CSMA large system ($M = 50$) with and without collision detection, and $B = 1$ buffer per station. The small loss experienced by C_1 in the cases shown is mainly due to the large number of buffers in the system and the longer interarrival times of packets to any given station. There is practically no difference in packet loss *with or without collision detection. This is encouraging for radio* applications where the collision detection feature is not implementable.

Finally, to study the effect that packet length has on system loss we show some results in Figure 27. The curves shown are those of a system with $M = 5$, $B_1 = B_2 = 1$, $T_1 = 100$, $T_2 = 10$ slots, $p_1 = p_2 = 0.2$ and $T_e = 2$. The longer packet length of C_1 causes a *smaller number of packets* to be transmitted for the constant throughput $S_1 = 0.1$. As less C_1-packets are transmitted, a smaller number of packets is lost. The similarity between the two modes of preemption is due to the same reasons explained in the previous subsection for this combination of packet lengths.

5.3 Variance of Delay

We wish to evaluate the variance of delay because, among other reasons:

a. It gives an indication of the fairness of the protocol,

b. It is very important that the variance of delay be kept low if a network is to support real time applications.

As with its average, the variance of packet delay increases as the throughput increases and, under a random access scheme, becomes very large as the total throughput approaches channel capacity. We inquire as to how priority functions can alleviate this

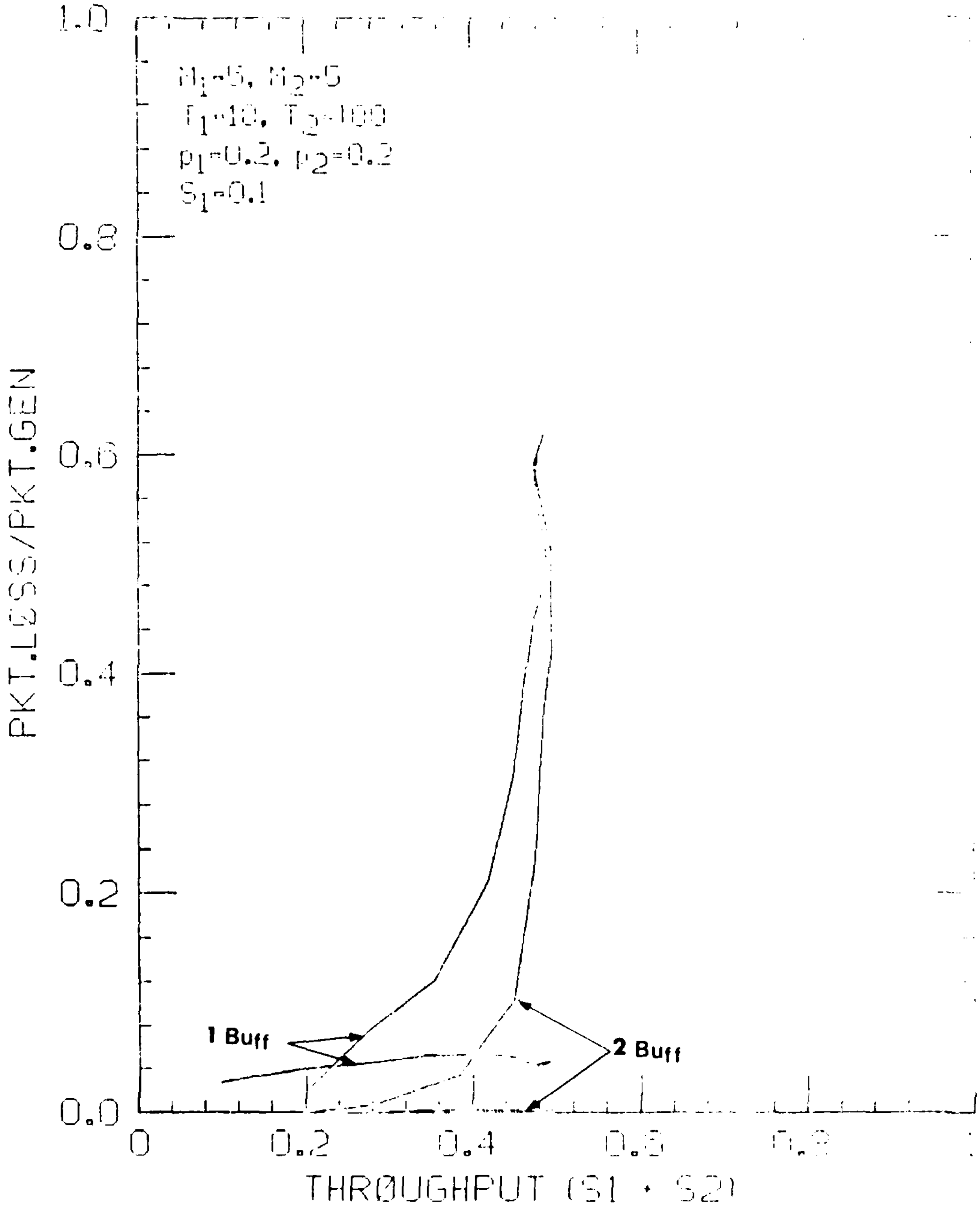


Figure 25. Packet loss in preemptive systems with 1 and 2 buffers per station

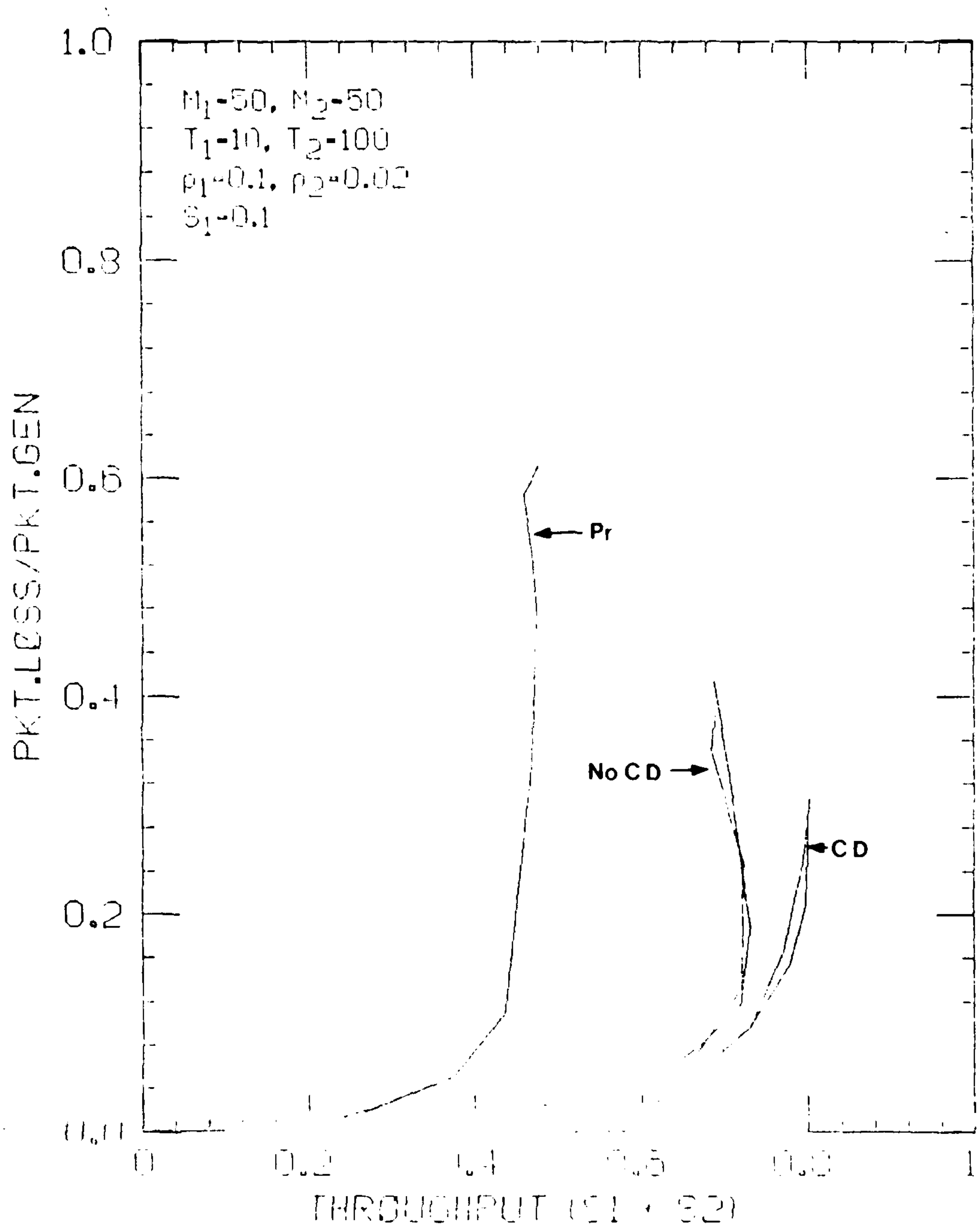


Figure 26. Comparison of packet loss in systems with and without collision detection

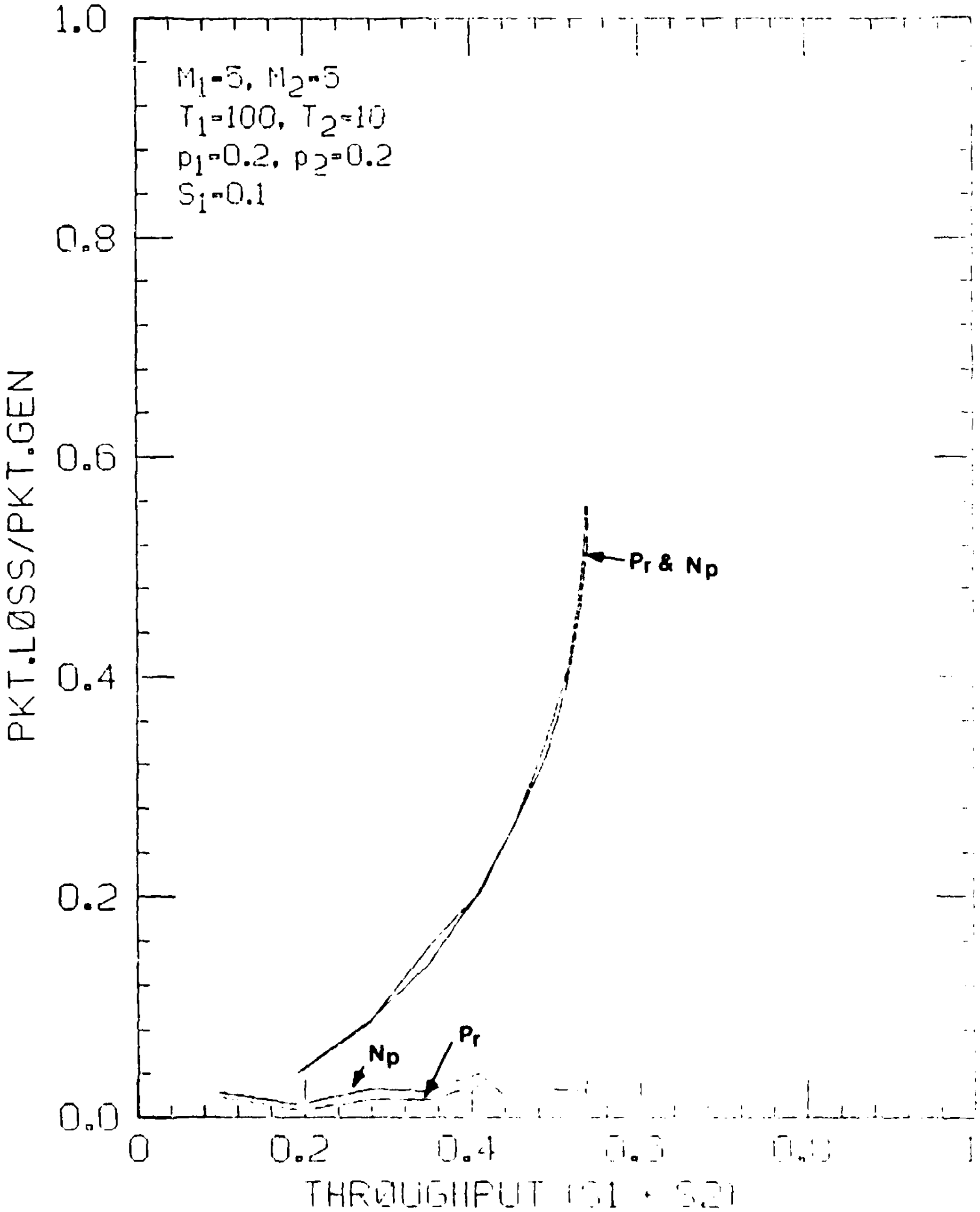


Figure 27 Packet loss in prioritized systems with long high priority messages

problem for high priority traffic despite the presence of high load in lower priority levels. Obviously in a prioritized environment we wish to minimize the variance of delay for the high priority class.

In Figure 28 we compare the variance of delay of C_1-packets in a nonpreemptive system to that of a nonprioritized system. The system under consideration has $M = 5$ stations, $B_1 = B_2 = 1$, $T_1 = 10$, $T_2 = 100$ slots, $p_1 = p_2 = 0.2$ and $T_r = 2$. We show the curves for $S_1 = 0.1$ and $S_1 = 0.2$. We note that the variance in the nonpreemptive case is smaller, at high throughput, than the variance shown for the nonprioritized case. More importantly, C_1-variance does not grow unbounded, thus ensuring good delay performance for C_1 packets over the entire range of throughput shown. The first point of every curve corresponds to the case when no C_2 packets are present. Note the sharp increase in variance as soon as some C_2 packets arrive at the system. This increase is explained as follows: when C_2-packets start arriving at the system, some C_1 packets will encounter ongoing C_2-transmissions upon arrival, having to wait for the next PAP to gain control of the channel. These C_1-packets will incur delays much larger than the average delay incurred by all other C_1 packets. This difference causes the sharp increase in variances from the first point in the curves to the next. The variance for C_2 packets, on the other hand, is larger in nonpreemptive systems than in nonprioritized systems, as we show in Figure 29.

As it was the case with packet delay and packet loss, the variance of delay, for a small system, is not too sensitive to variations in $p_i, i = 1, 2$ as we show in Figure 30. In the case of a large system however, the choice of near optimum $p_i, i = 1, 2$, has a more significant effect. This is shown in Figure 31 for a nonpreemptive discipline. In fact the sensitivity to the choice of $p_i, i = 1, 2$, for C_1 is larger when the load is lower because the contention period for C_2 is then expected to be larger. This causes larger delays for C_1 packets that arrive after the begining of the C_2 contention period.

Just as with average packet delay increasing the number of buffers in the system does not affect the variance of delay for C_1 packets, as we show in Figure 32. The variance of C_2, however, increases sharply when the system has two buffers due to the larger queueing delays incurred by packets which occupy the second buffer.

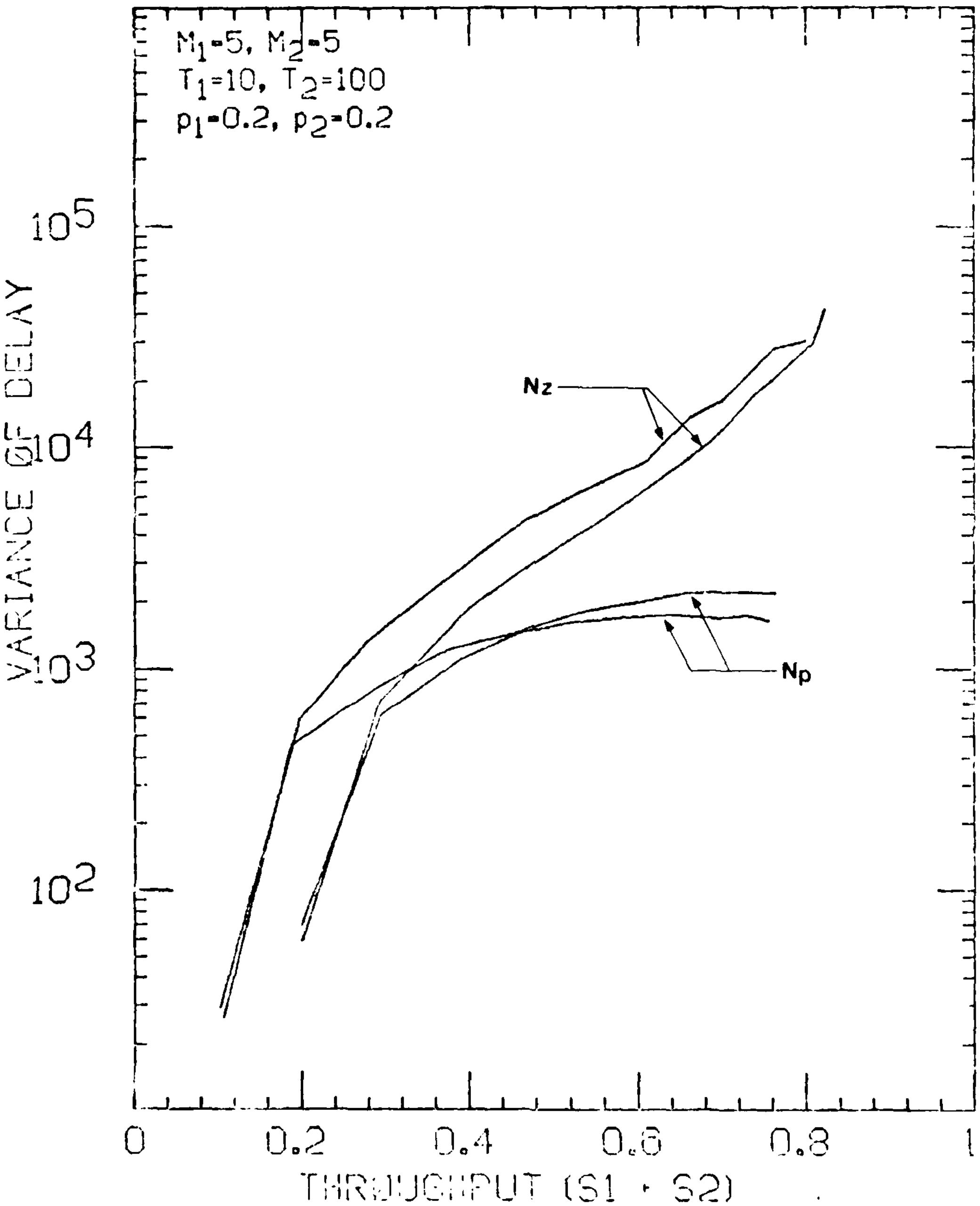


Figure 28. Variance of Delay in prioritized and nonprioritized systems for C_1 messages

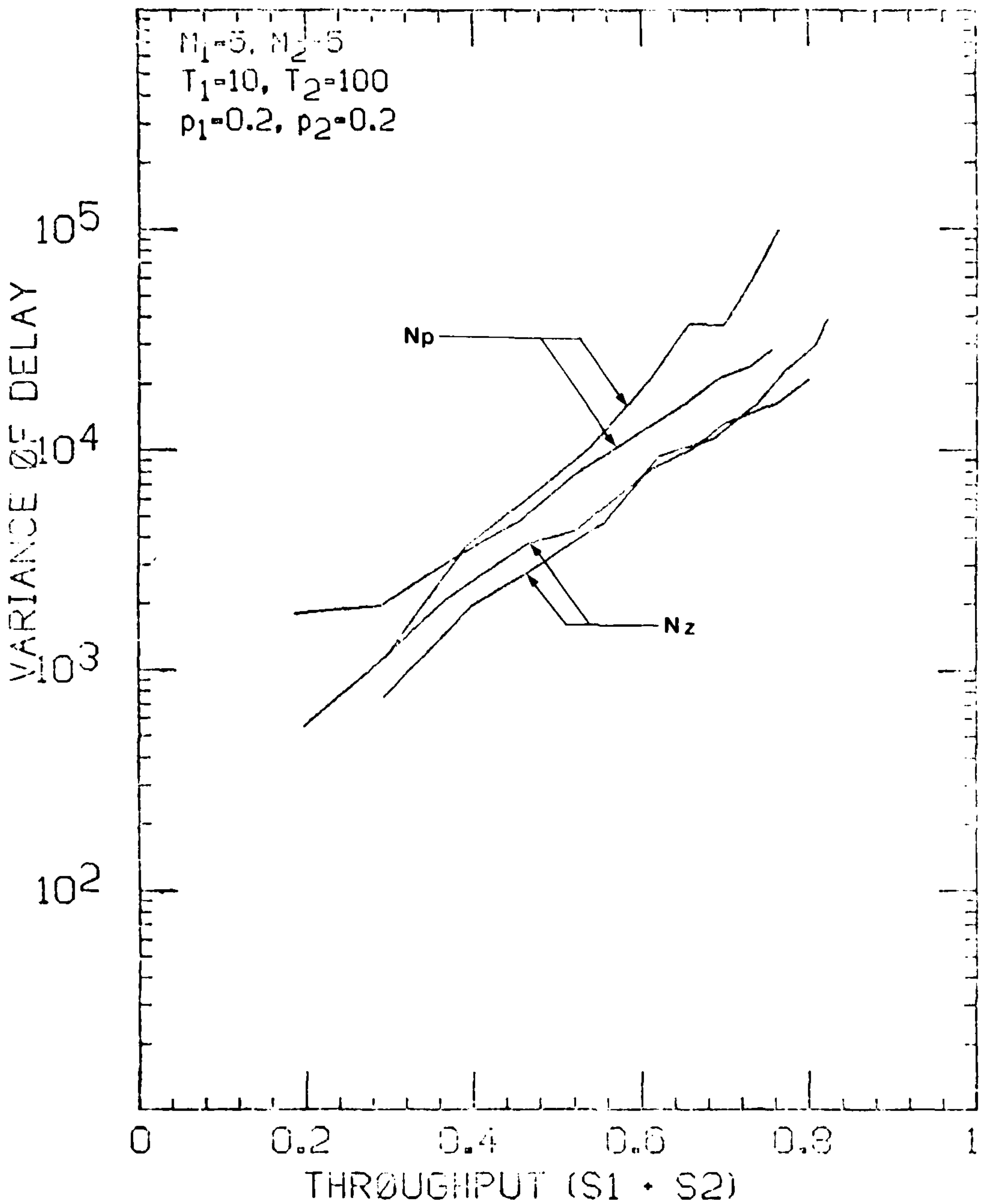


Figure 29. Variance of Delay in prioritized and nonprioritized systems for C_2 messages

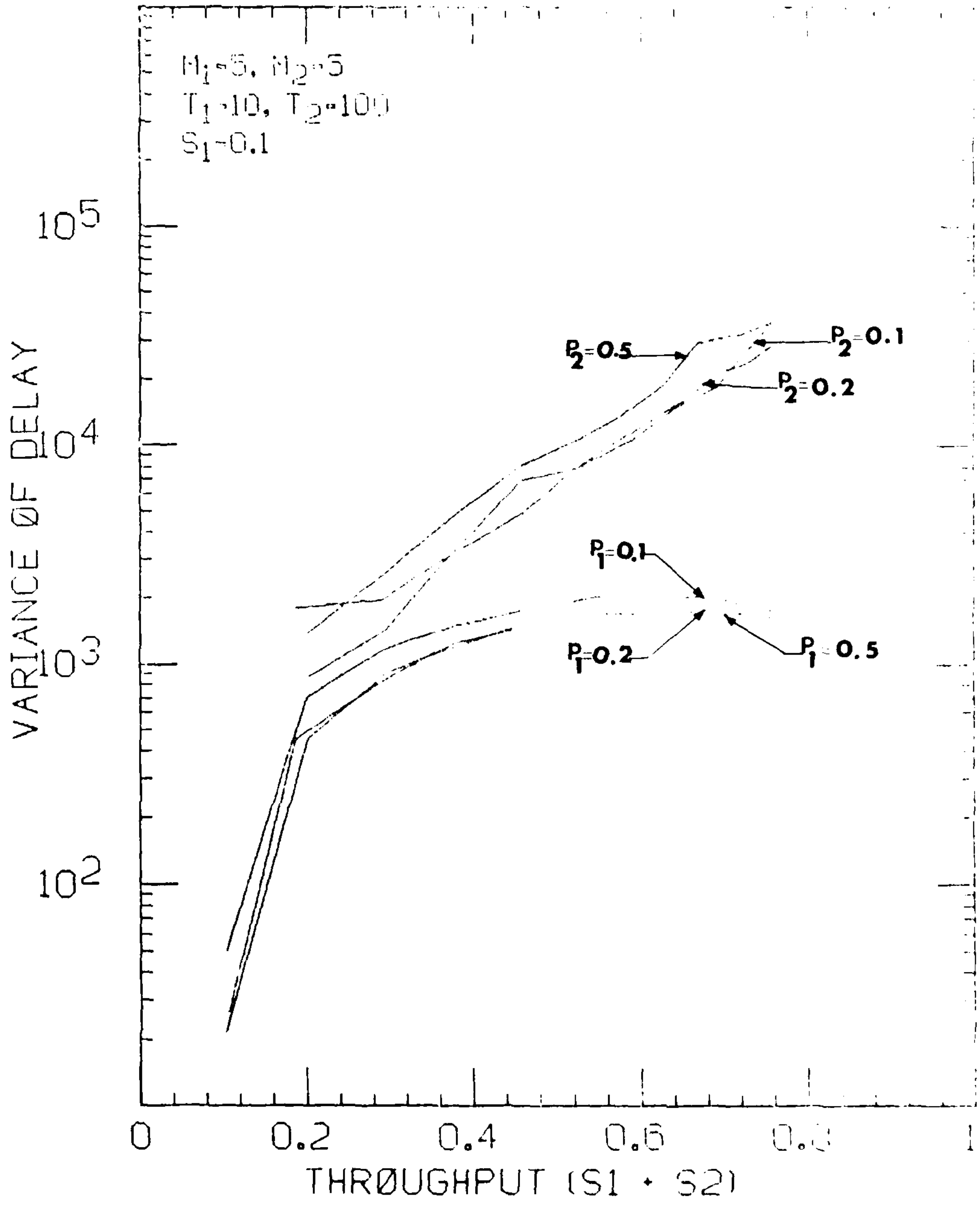


Figure 30. Variance of delay in a nonpreemptive system with different values of p (0.1,0.2,0.5)

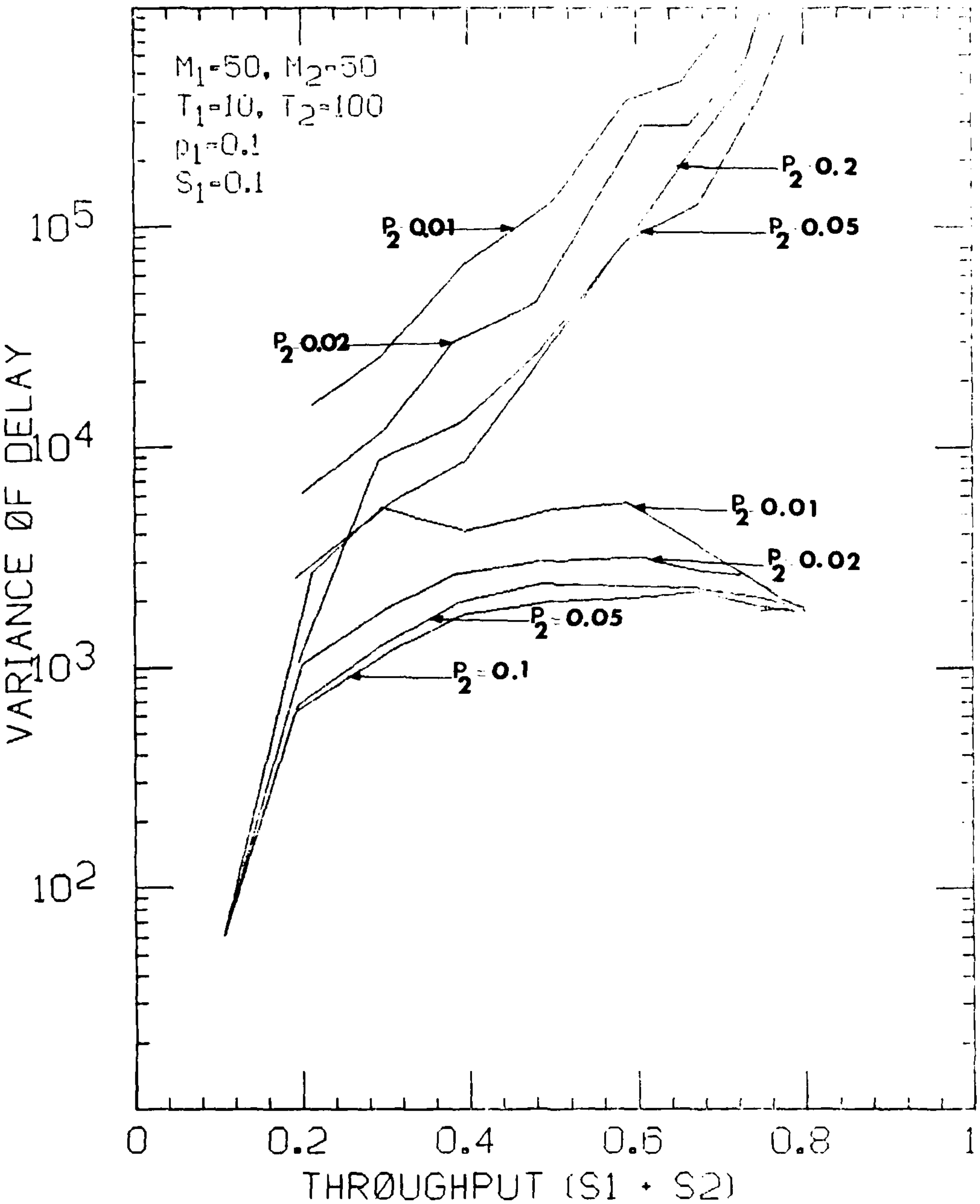


Figure 31. Variance of delay in large (50 stations) nonpreemptive systems at different values of p (0.01,0.02,0.05,0.1)

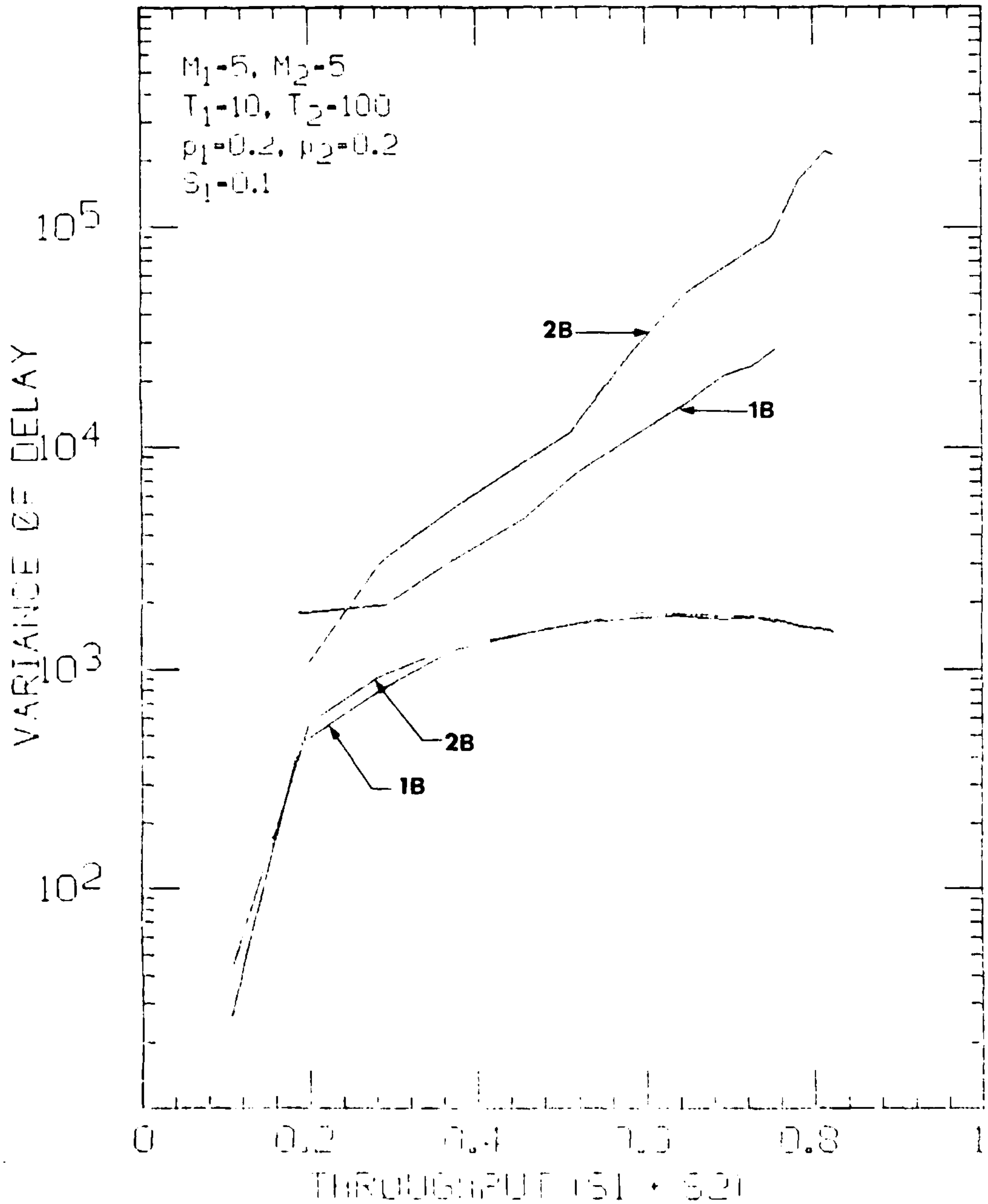


Figure 32. Comparison of the variance of delay in nonpreemptive systems with 1 and 2 buffers per station

We now turn our attention to the effect that preemption has on the variance of delay. This effect is shown in Figure 33 for C_1 and Figure 34 for C_2, for small systems. We first note the similarity between systems in the nonpreemptive mode and semipreemptive mode ($T_p = 0$). As we increase T_p, the variance of C_1 packets decreases. This is due to the fact that a larger fraction of C_1-packets which arrive to the system during an ongoing C_2-transmission can preempt that transmission. Thus both the average delay and the variance are decreased. C_2, however, shows the opposite effect. As we increase T_p, the probability that a transmission is preempted increases, thus packets incur larger delays and variances. The total effect of preemption on the variance of delay is summarized in Figure 35, where we plot the variance vs. the fraction of T_2 in which preemption is allowed and we observe the effects just described. Note that close to full preemption the total C_1-variance is less than 60 $(\text{slots})^2$ making it very reliable for applications with delay constraints.

Collision detection does not have a significant effect on the variance of delay. In Figure 36 we compare the variance of delay in large systems with and without collision detection in the nonpreemptive and semipreemptive mode. A fully preemptive system with collision detection is included in the figure for comparison purposes. We can observe that systems operating in equal modes show very similar behavior regardless of whether the collision detection feature is in effect or not. Note also that in these large systems, the difference in the variance of delay between semipreemptive and nonpreemptive modes is significant.

5.4 A Special Case: Generation at Fixed Intervals

So far all the results presented are for systems in which the generation process for both priority classes has a geometric distribution of interarrival times. One of the applications that local networks may support is digitized voice transmission, which presents traffic characteristics that are different from the ones considered so far. While the study of voice transmission is presented in the next section, we present here, as an application exercise, a special case of generation process of high priority messages. We consider that

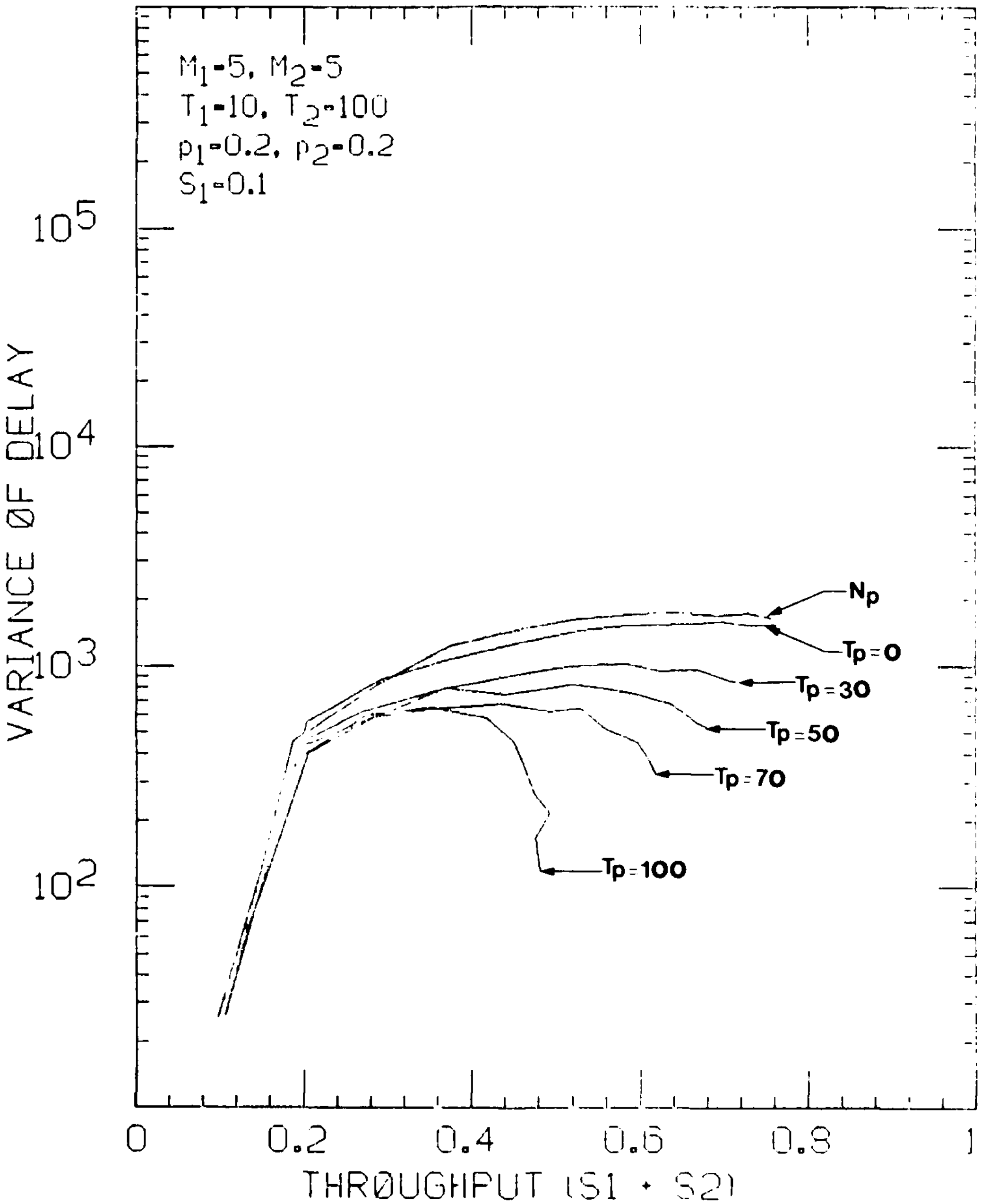


Figure 33. Variance of delay for C_1 messages in preemptive systems at different levels of preemption

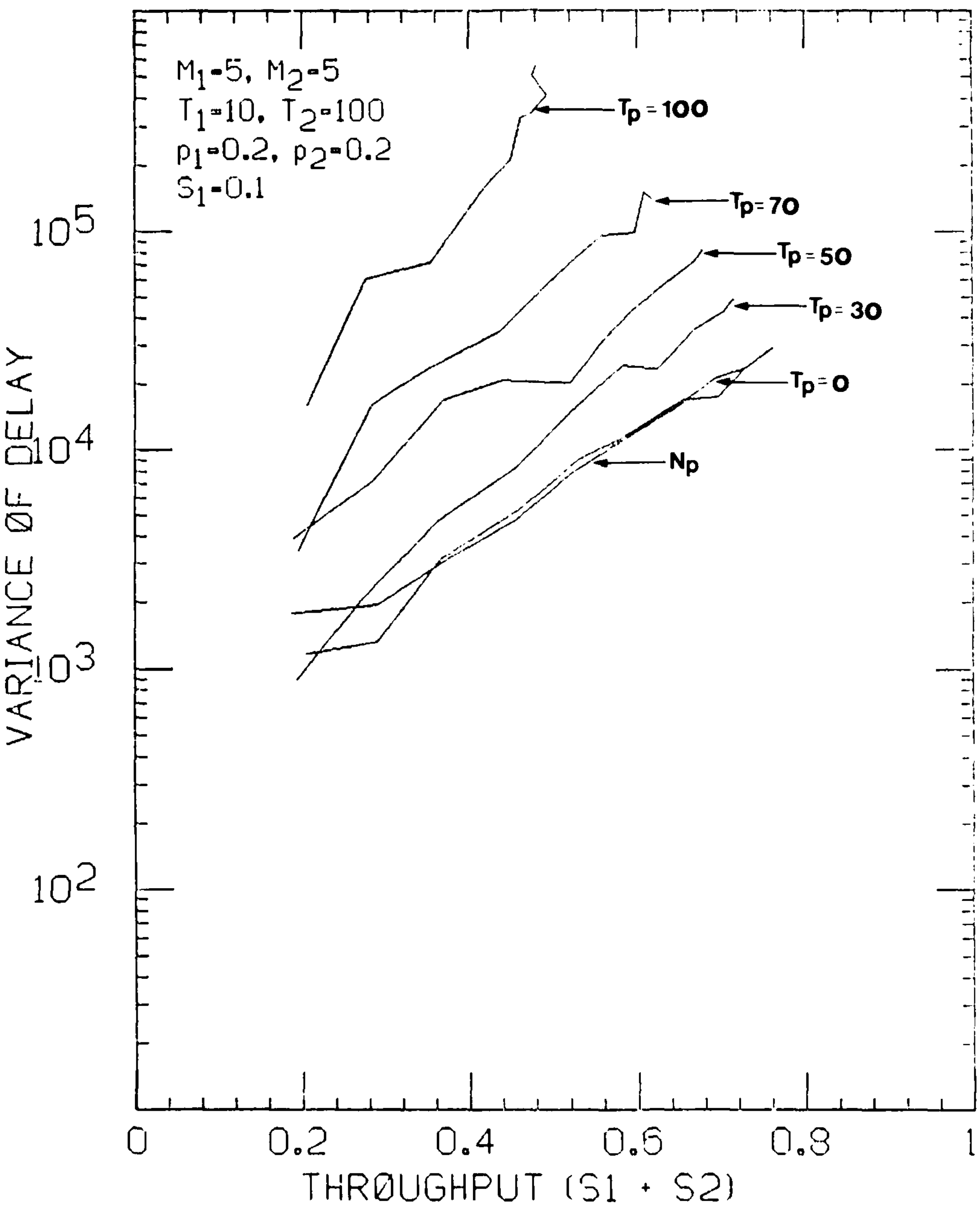


Figure 34. Variance of delay for C_2 messages in preemptive systems at different levels of preemption

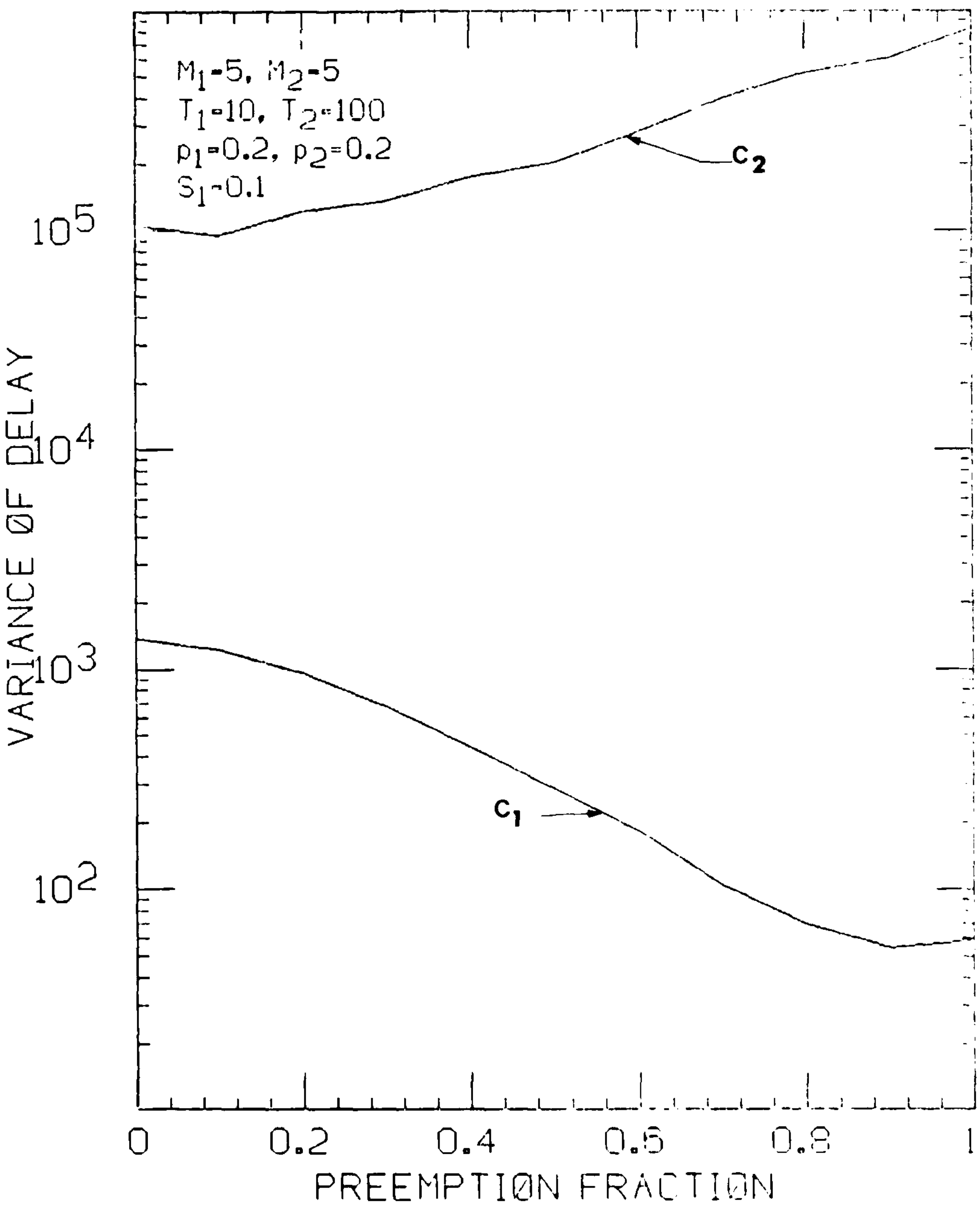


Figure 35. Variance of delay as a function of the preemption fraction

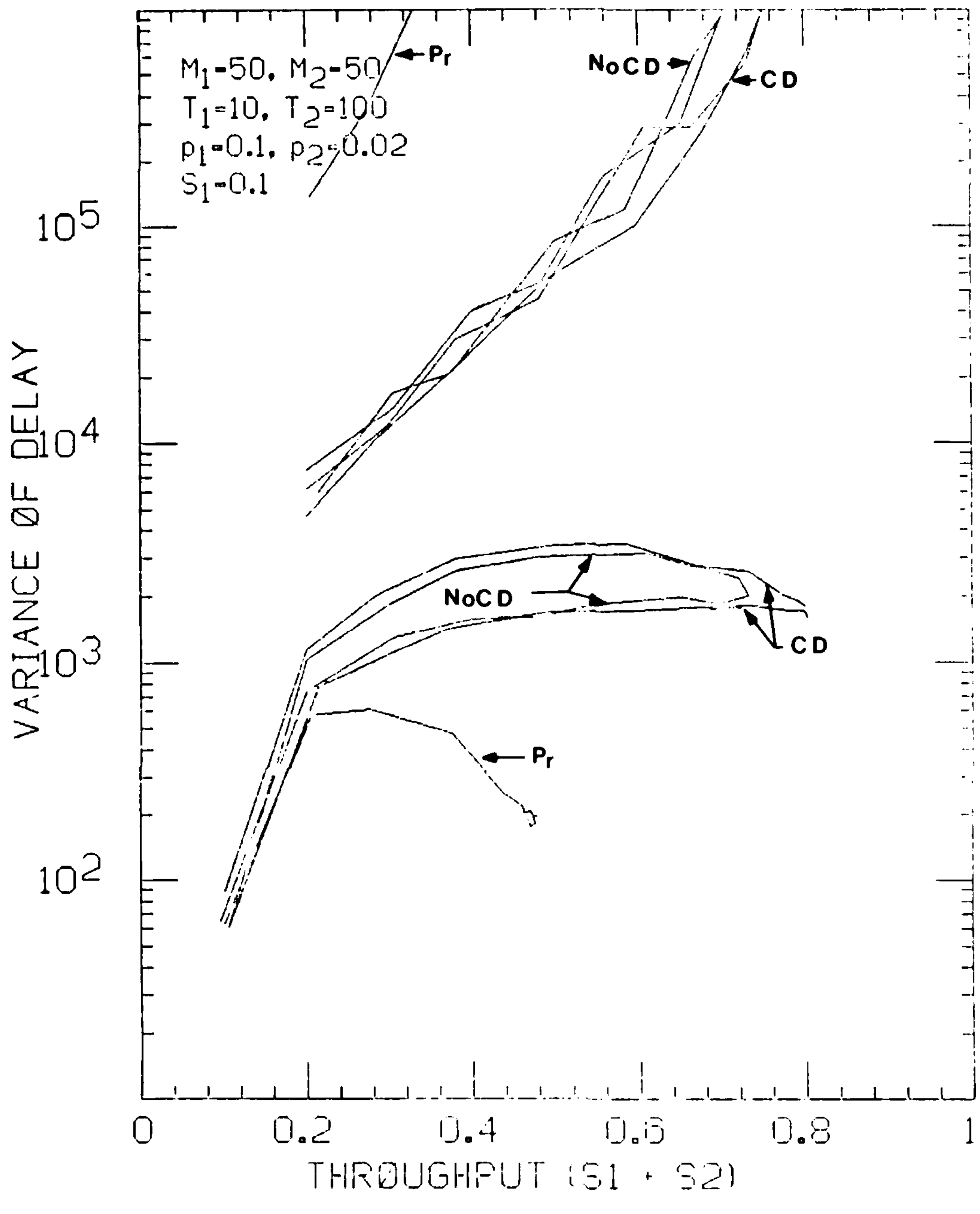


Figure 36. Comparison of the variance of delay in systems with and without collision detection

these messages are generated at each station at the rate of 64000 bits/sec, as it could be the case in pulse code modulation for voice transmission (PCM). The *transmission medium* is a coaxial cable 1000 m long, in which the propagation delay is ns/m, giving an end-to-end propagation delay of $\tau = 10\mu s$; the channel bandwidth is 10 Mb/sec and the packet size for C_1 messages is 1000 bits (the transmission time is, therefore, 10 slots). The system has $M = 50$ stations and each station generates packets every 0.015625 sec or every 1562.5 slots, causing the throughput of C_1 to be $S_1 = 0.32$, which is a heavy *load for the high* priority class. Low priority messages are generated as before, from a geometric distribution with parameter σ_2 and are 100 slots long.

Several *interesting* results were obtained by simulating this case. We considered that the arrival of the first packet generated at each station (for system startup) occurred at a point in time *uniformly distributed in the interval* [0 , 1562.5] slots. From there on, a station generates one packet every 1562.5 slots. Figure 37 shows the packet delay for high priority messages; the delay for low priority messages ranged from 1049 slots to 25000 slots and thus is not shown in the figure. It is important to note that for different values of σ_2 (and thus S_2) the starting point of each station is again selected from a uniform distribuition in the interval [0 , 1562.5]; this can explain the irregular behavior that the curve is exhibiting. Indeed if the pattern is such that there are many arrivals grouped in a *small period of time*, then the contention caused and the resolution required increase the delay even if the load is not increased.

It may be worthwhile to point out here that in an application of this type, full *preemption may* be a *mistake*. For the example under consideration, one C_1 packet arrives to the system on the average every 30 slots. With preemption a C_2 message of 100 slots has a high probability of being preempted and thus no C_2 packet will *ever be able to* get through. A simulation of such a case was performed and the total throughput delay characteristic of the system was a series of points clustered together around $S = S_1 = 0.32$. The throughput for C_2 was indeed $S_2 = 0$ indicating a very large amount of thrashing on the channel. Therefore, in this application, a very important factor in determining system *performance* is the distribution of arrivals within a cycle.

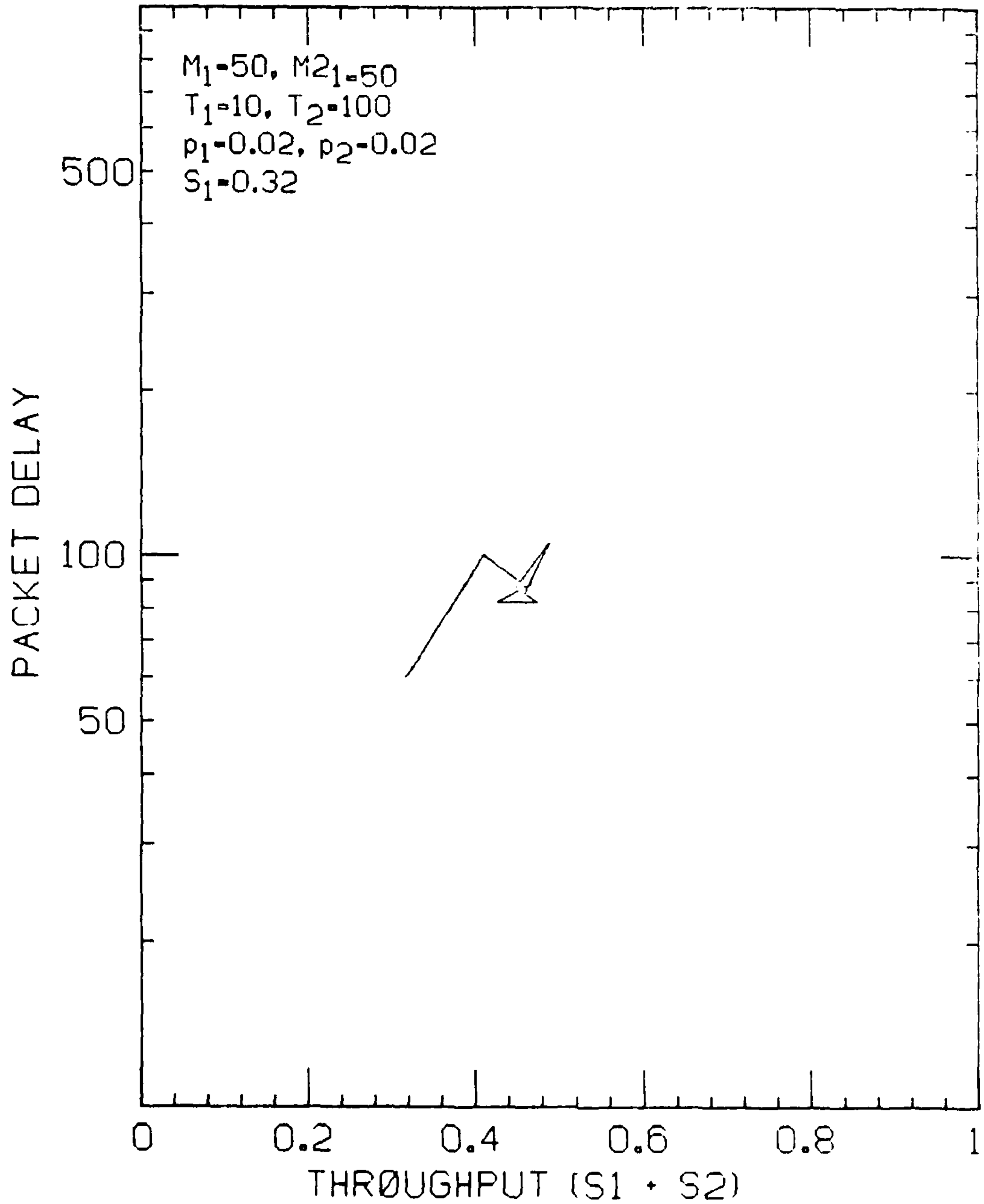


Figure 37. Packet delay for C_1 messages when their generation is at fixed intervals

In contrast with the above results, nonprioritized systems exhibit a much smoother response, regardless of the pattern of first arrivals from the independent deterministic process as is shown in Figure 38. Note however, the large increase in packet delay for short messages in the nonprioritized system. In this figure we also include the throughput delay characteristics of prioritized systems in the nonpreemptive mode with and without collision detection.

The best performance of prioritized systems in this special case, is with respect to packet loss. In all the simulations performed not a single C_1 packet was lost. S_2 however, was very small ($<$ 16% in the best case). In nonprioritized systems packet loss was also low as long as $S < 0.7$ (i.e., $S_2 = 0.38$), as it is shown in Figure 39. But clearly in this case the average delay and the variance of delay increase as S increases from $S = S_1 = 0.32$ to $S = 0.7$. The variance of delay is shown in Figure 40.

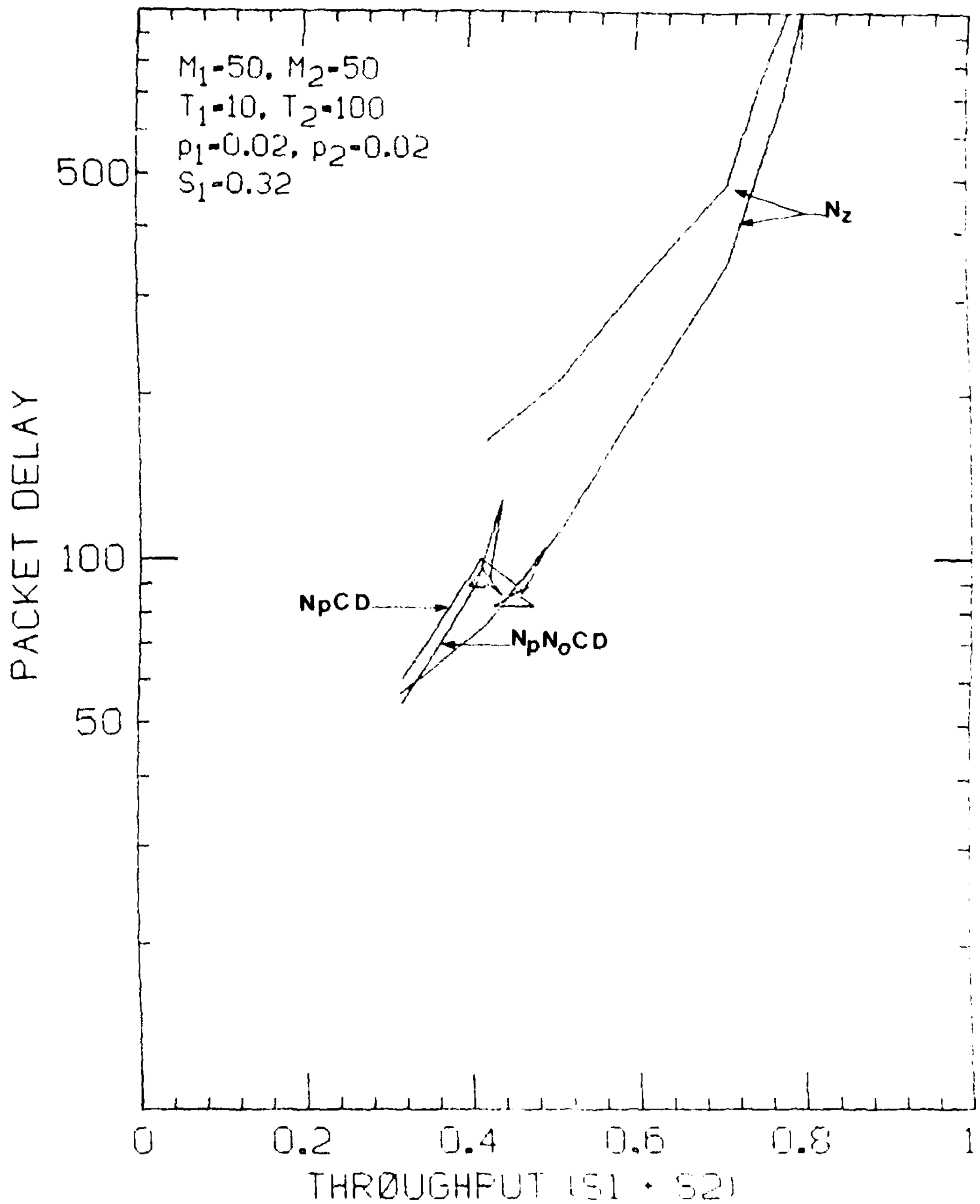


Figure 38. Packet delay of prioritized (with and without collision detection) and nonprioritized systems with fixed generation of C_1 packets

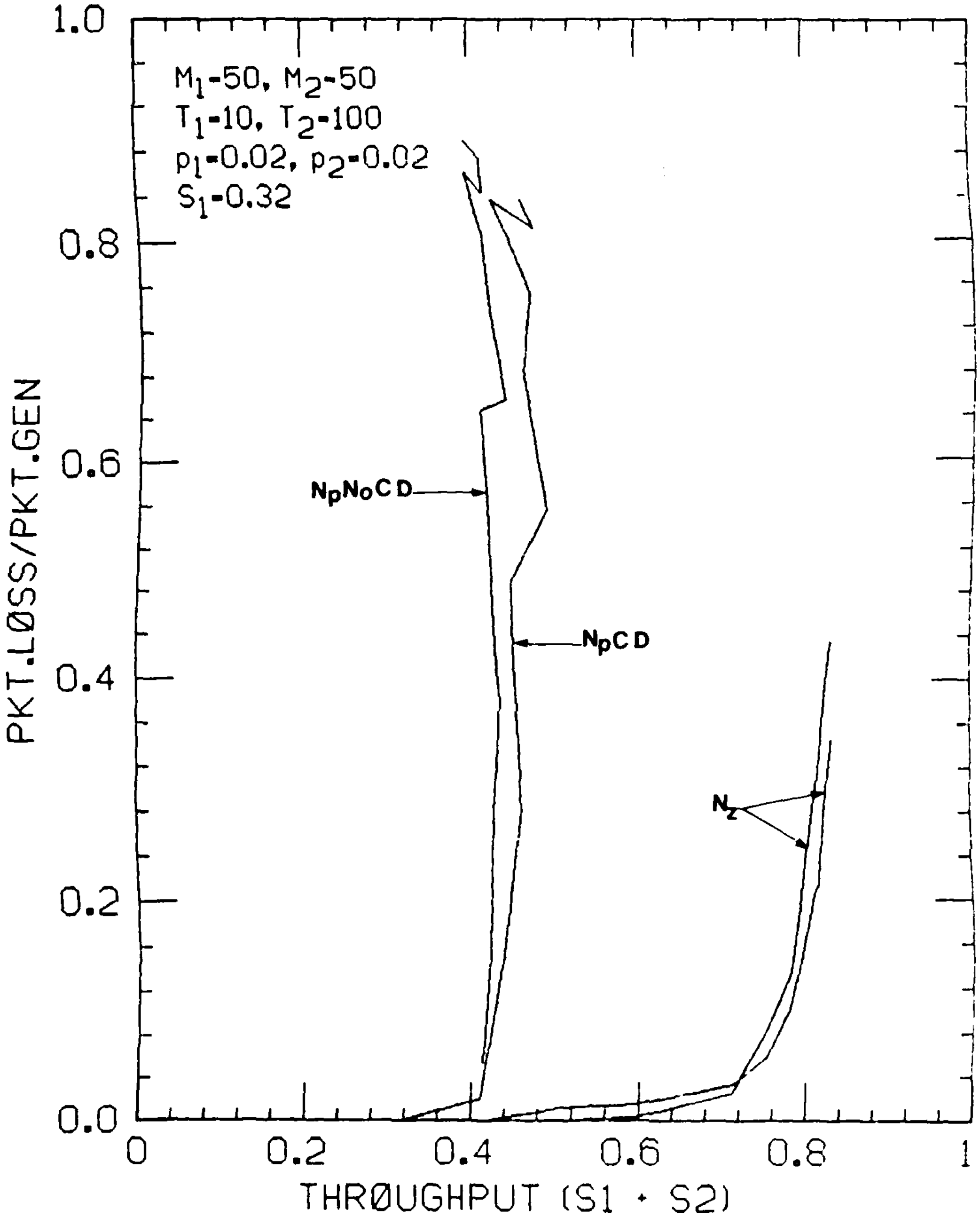


Figure 39. Packet loss of prioritized (with and without collision detection) and nonprioritized systems with fixed generation of C_1 packets

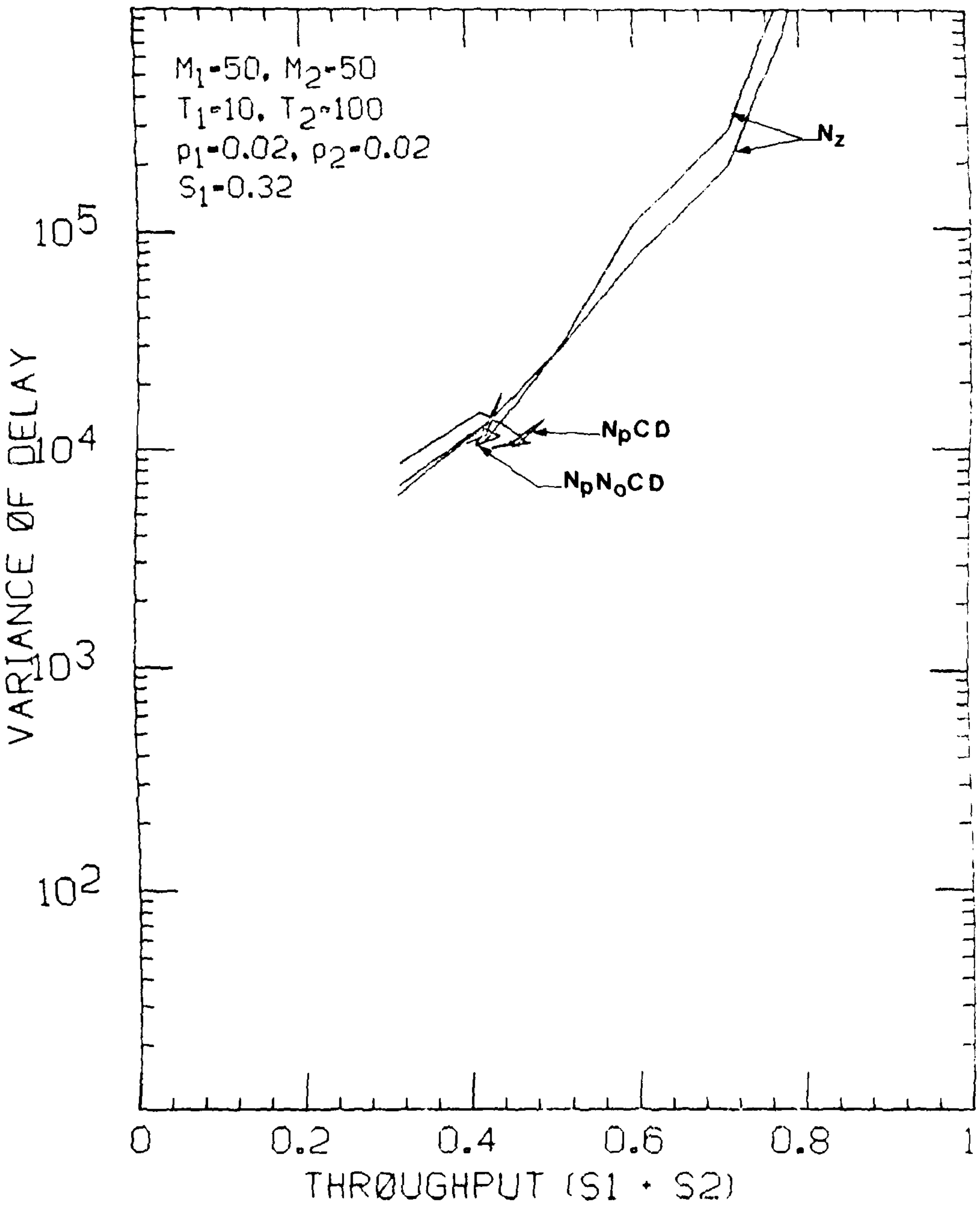


Figure 40. Variance of delay in prioritized (with and without collision detection) and *nonprioritized* systems with fixed generation of C_1 packets

§6 Voice Transmission in P-CSMA Networks

Recently, the desire to integrate voice communication on local data networks has been apparent. In this section we investigate the performance of our systems when supporting voice communication. In particular we study the effect on performance of various system parameters, such as channel bandwidth, vocoder rate, delay requirement, allowable packet loss rate, etc. For comparison purposes, we also consider an ideal conflict free TDMA case which is undoubtedly the most sitable for voice traffic exhibiting a deterministic generation process, and thus provides the ultimate performance one can achieve.

6.1 Packet Generation Process

In most of the previous sections it was assumed that for each user the packet intergeneration time is a random variable with a memoryless distribuition (discrete time geometric, with the time unit equal to τ sec.). When dealing with voice applications, such an assumption is not adequate as the packet generation process is to a first approximation deterministic (see below). Moreover, due to the real time constraints encountered in voice communication, average performance is not sufficient, and one has to derive the distribuition of delay or delay percentiles. Typically, if the intent is to integrate voice and data on the same network, then, due to the strict end-to-end delay requirement in voice applications, one suspects that the prioritized scheme would be more appropriate. Indeed by giving priority to voice packets over data packets, the scheme will help guarantee to a certain extent the delay constraint for voice packets even on the presence of data traffic. Note, however, that in the present section, we consider that there exists only one class of traffic, namely voice, and that it is given the highest priority.

6.2 Characteristics of Voice Traffic and Voice Sources

It is assumed that vocoders digitize voice at some constant rate of V bits per second. Bits are grouped to form packets which are then transmitted via the network to

the destination vocoder. Let B denote the number of bits per packet. B is the sum of two components: B_h which encompases all overhead bits comprising the preamble, the packet header and the checksum, and B_v, the information bits. The time to form a packet, T_f, is given by

$$T_f = \frac{B_v}{V}$$

T_f is also the packet intergeneration time for a vocoder. To achieve interactive speech and smooth playback operation, it is important to keep the end-to-end delay for most bits of voice information within tight bounds. End-to-end delay is defined as the time from when the bit is generated at the originating vocoder until it is received at the destination vocoder. Two components of delay are identified: the packet formation delay, T_f, and the packet network delay D_n. The network delay is defined as the time since the packet is formed until it is successfully received at the destination. Denoting by D_m the maximum allowable delay for voice bits, a voice packet is acceptable only if $T_f + D_n \leq D_m$. Packets which do not satisfy this inequality are assumed to be lost. Usually speech can be effectively synthetized at the destination if the rate of lost packets does not exceed a maximum L. In voice applications, the performance measure is defined as the maximum number of vocoders that can be supported by the network under the delay constraint D_m and a tolerable loss rate L.

We assume that each voice source possesses a transmit buffer with room for exactly one packet. Whenever this buffer is nonempty, the station attempts transmission of the packet on the channel according to P-CSMA. We furthermore assume that if the buffer is nonempty when a new packet is generated, then the former is lost and the latter occupies the buffer (i.e., the order of service is last come-first served). Although this model appears to be restrictive a priori, we shall show in the following subsection that this is not so. In fact for a given delay requirement D_m, the optimum packet size which maximizes the number of voice sources is such that $D_n < T_f$ for all values of L. That is, at optimum we have $D_m < 2T_f$, and therefore there is no need for a transmit buffer of size larger than one, and LCFS is the appropriate queueing discipline. At optimum, packet loss contributing to L is only due to excess delay and not to lack of buffers.

Before we proceed with the discussion of the numerical results obtained from the simulation of P-CSMA, we undertake here an idealized analysis which provides an upper bound on the performance. In essence it consists of assuming that network delay D_n is ideally deterministic and equal to only the transmission time on the channel of bandwidth W. With this assumption, $D_n = B/W$ and the condition $T_f + D_n \leq D_m$ leads to

$$\frac{B_v}{V} + \frac{B_h + B_v}{W} \leq D_m \tag{1}$$

and thus

$$B_v \leq \frac{(W D_m - B_h)V}{(V + W)} \tag{2}$$

Given that M voice sources are active, the bandwidth constraint is then written as

$$W \geq MV \frac{(B_v + B_h)}{B_v} \tag{3}$$

Eqs. (2) and (3) lead to a maximum value of M given by

$$M^* = \frac{W D_m - B_h}{V D_m + B_h} \tag{4}$$

This ideal analysis in fact corresponds to TDMA in which perfect synchronization is achieved; i.e., the voice packet for a user is ready for transmission exactly at the beginning of the slot assigned to that user. This is simply achieved by having the vocoder synchronize the formation time of the packet with the boundary of its assigned slot. Therefore with M users and TDMA frames of M slots, D_n is equal to *one* slot (i.e., $(B_h + B_v)/W$), while T_f is equal to M slots; hence equations (1) and (3).

Equation (4) illustrates the effect of the overhead B_h on M^*. If $B_h = 0$, then $M^* = W/V$, independent of D_m. In that case however the optimum packet size is a function of D_m, given by

$$B_v = \frac{D_m}{\left(\frac{1}{V} + \frac{1}{W}\right)}$$

The smaller D_m is, the smaller the packet size is. With $B_h \neq 0$, M^* is a function of D_m, and decreases as D_m decreases. Indeed, with $B_h \neq 0$, the packet size cannot be arbitrarily decreased, as the effect of overhead becomes more severe. In Figures 13 and 14 the dashed

curves represent M^* as a function of V for $W = 1$ MBPS and $W = 10$ MBPS respectively, $B_h = 200$ bits and various values of D_m. $D_m = 200$ msec. corresponds roughly to the case $B_h = 0$ or $D_m \to \infty$. For $D_m = 2$ msec., the effect of B_h is so important that M^* is limited to very small values and is rather insensitive to V.

6.3 Discussion of Numerical Results

We consider a P-CSMA network, 1 Km. long with an end to end propagation delay τ of approximately $10 \mu sec$. We assume $B_h = 200$ bits which accounts for a 64 bit preamble, a 32 bit CRC and 104 bits for addressing and other control information. We consider various values for the vocoder bandwidth: $V = 16$, 24, 32 and 64 kbps; and two values for the channel bandwidth: $W = 1$ and 10 MBPS.

The delay constraint on voice bits depends on the type of voice communication being considered. The first type we identify is that of real time voice communication within the local environment (i.e., all parties are on the local network); in this case interactive communication can be effectively accomplished if D_m is on the order of 200 msec. (or even higher). The second type is that of real time voice communication via the Public Switched Telephone Network (i.e., where all parties are not within the local network); in this case D_m must be restricted to a value on the order of 20 msec. Finally we distinguish a third type wich arises when four wire terminations and echo cancelling procedures are not available; in this case the echo problem can be so severe as to require D_m to be limited to very small values, on the order of 2 msec. As previously mentioned with voice communication, packet loss is tolerable as long as the loss rate L is limited to a small value. In this study we shall assume that $L = 0.02$ is adequate.

We first examine the effect of packet size $B_h + B_v$ on the delay performance. Consider W, V and M to be fixed. Let K denote the number of packets sampled in the simulation, and let $D_1, D_2, ..., D_K$ be the delay incurred by the K packets respectively. Let $D_{i_1} < D_{i_2} < ... < D_{i_K}$ be the ordered sequence of delay samples. We let $\max\{D|L\} \stackrel{\Delta}{=} D_{i_{K(1-L)}}$; $\max\{D|L\}$ is the value of packet delay which is exceeded by exactly

a fraction L of all samples. We can similarly define $\max\{D_n|L\}$. Clearly $\max\{D|L\} = T_f + \max\{D_n|L\}$. If B_v is arbitrarily large, then T_f is the predominant term. Indeed, the number of packets contending is small and with the packet size being large, the performance of P-CSMA (which is a function of $\tau W/B$) is relatively good. In this case $\max\{D|L\}$ is also larger, on the same order of magnitude as $B_v/V + B/W$. As B_v decreases, T_f decreases and so does $\max\{D|L\}$ until B_v is small enough as to cause a high degree of contention and an important decrease of $\max\{D|L\}$. Clearly, further decrease in B_v causes the P-CSMA channel to saturate and $\max\{D|L\} \to \infty$ with probability one. Thus there exists an optimum value for B_v which minimizes $\max\{D|L\}$. We illustrate these effects in Figure 11 in which we plot $\max\{D|L\}$ versus B_v for the case $V = 24$ KBPS, $W = 1$ MBPS, $M = 10$ and various values of L. It is interesting to note that given M, the optimum packet size B_v is roughly the same for all values of L. Setting $L = 0.02$, we show in Figure 12 $\max\{D|0.02\}$ versus B_v for various values of M. This Figure shows that for a given value of D_m and a given value of L, there exists a unique optimum value of B_v and a maximum value of M which satisfy the constraint $\max\{D|L\} \leq D_m$. This maximum value of M, M_{max}, represents the performance of P-CSMA when supporting voice communication.

From Figures 11 and 12 as well as the results obtained for all other cases studied, we note that $\max\{D_n|L\}$ at optimum is always *inferior to* T_f, and therefore $\max\{D|L\} < 2T_f$, regardless of D_m and L. This clearly justifies that the model adopted for the vocoder's transmit buffer (single packet buffer and LCFS) is not restrictive. In Figure 13 we plot M_{max} as a function of the vocoder rate V for $L = 2\%$ and $W = 1$ MBPS and the three values of D_m: 2 msec., 20 msec. and 200 msec. The dashed curves correspond to the ideal TDMA case. Figure 14 displays similar results for the case $W = 10$ MBPS. We note that when $D_m = 200$ msec., both M^* and M_{max} decrease rapidly as V increases; while if $D_m = 2$ msec., then M^* and M_{max} are rather insensitive to V. This is due to the existence of a nonzero overhead B_h, whose effect is more important as the delay reqirement is more critical. To best compare the performance of P-CSMA to that of the ideal TDMA, we consider the ratio M_{max}/M^*. This ratio has the property of isolating the effect of contention as opposed to that of V and B_h, and therefore allows us to evaluate

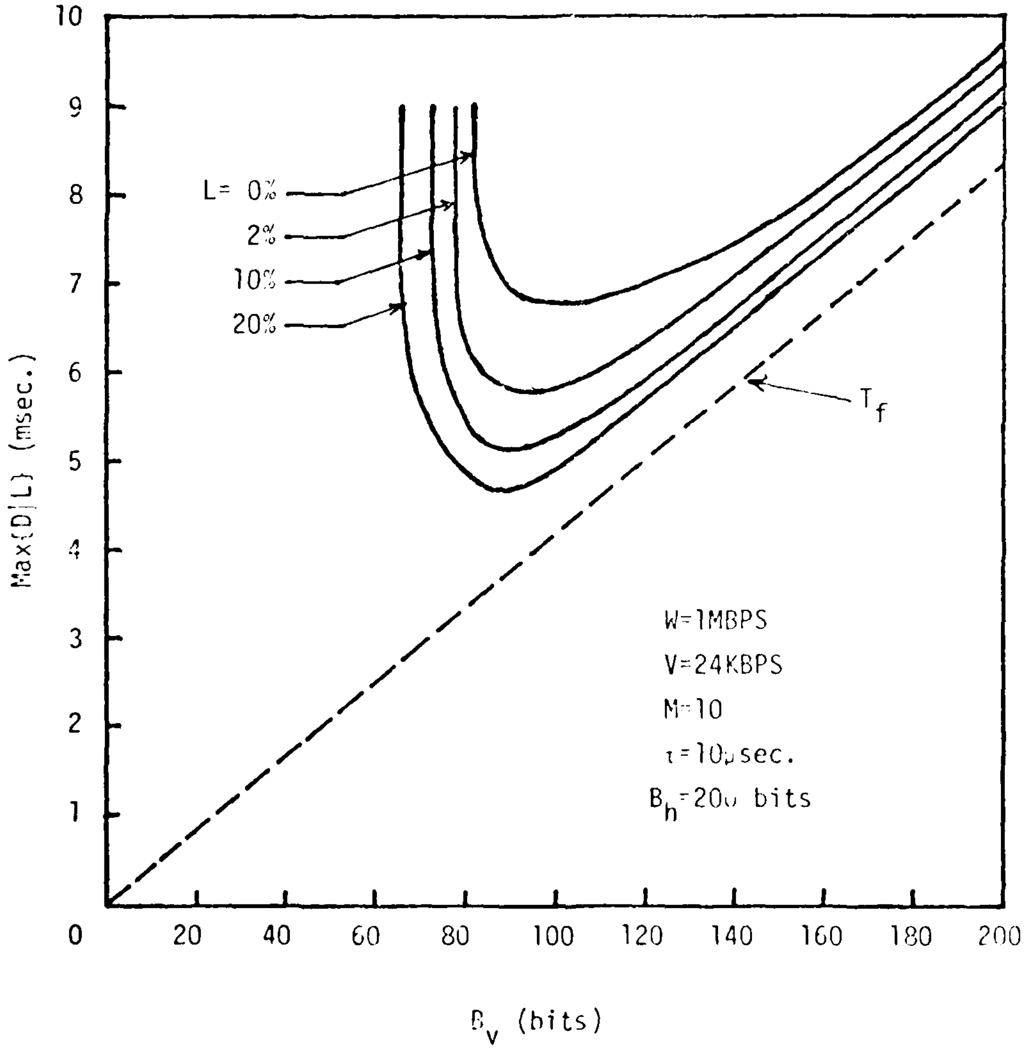


Figure 41 Max$\{D|L\}$ versus B_v for fixed M and various values of L

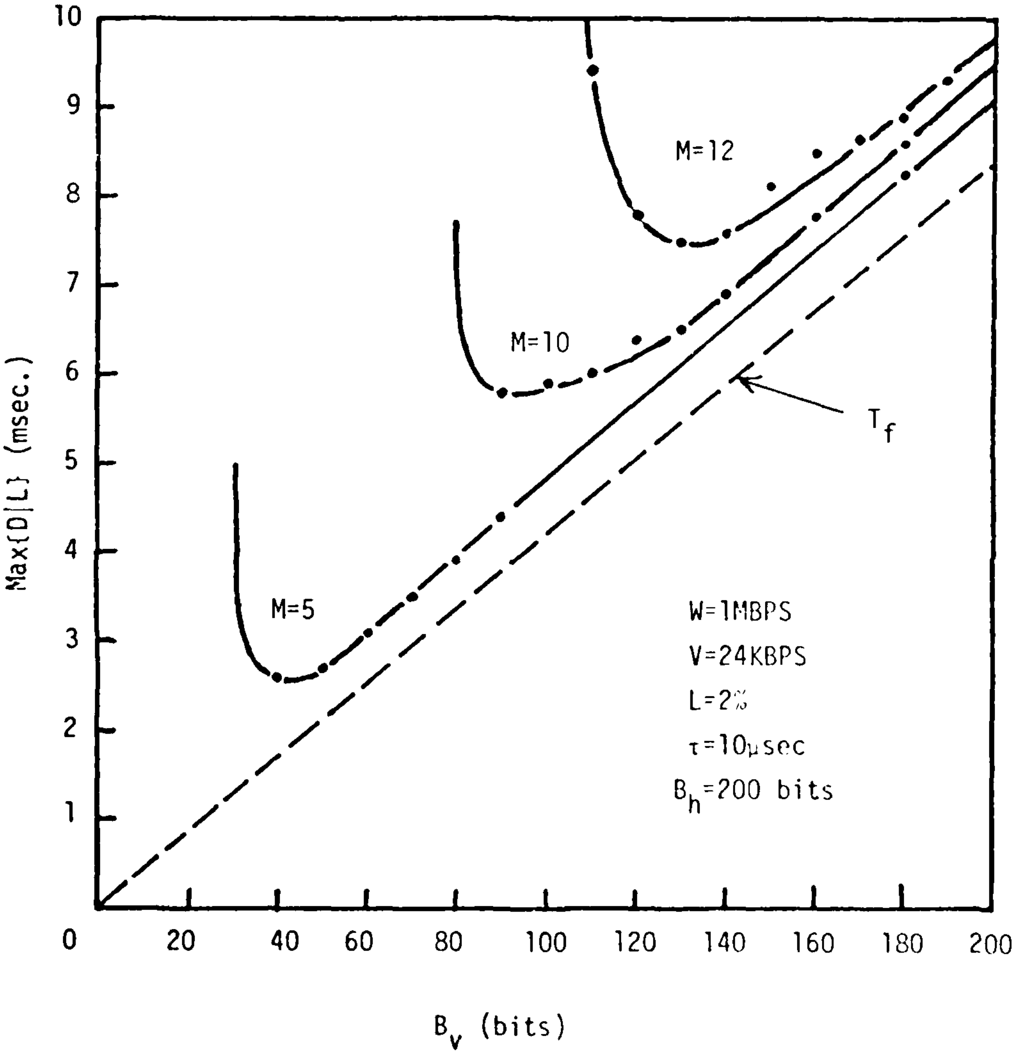


Figure 42. Max$\{D|L\}$ versus B_v for fixed L and various values of M

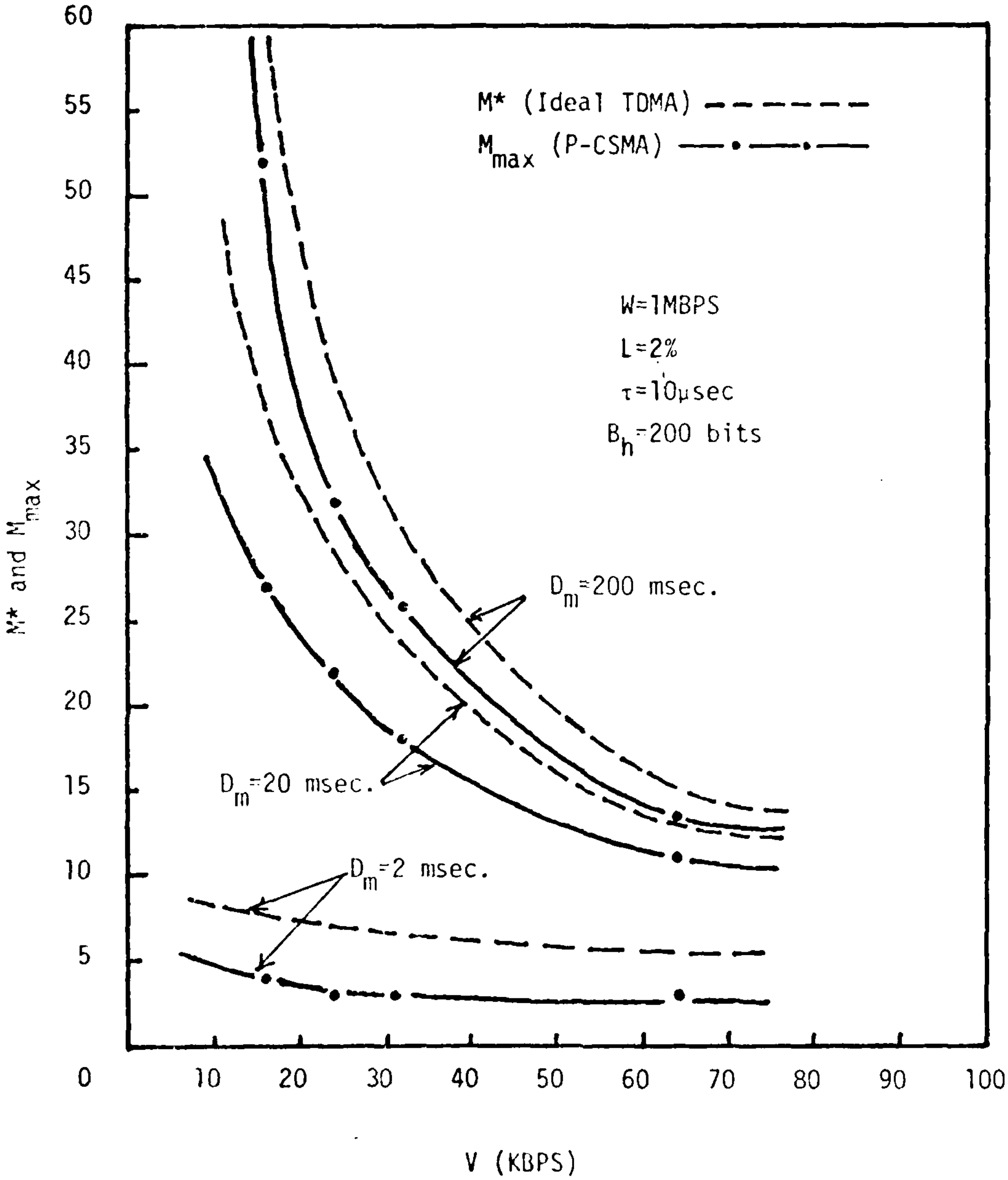


Figure 43 M_{max} and M^* versus V for $W = 1$ MBPS

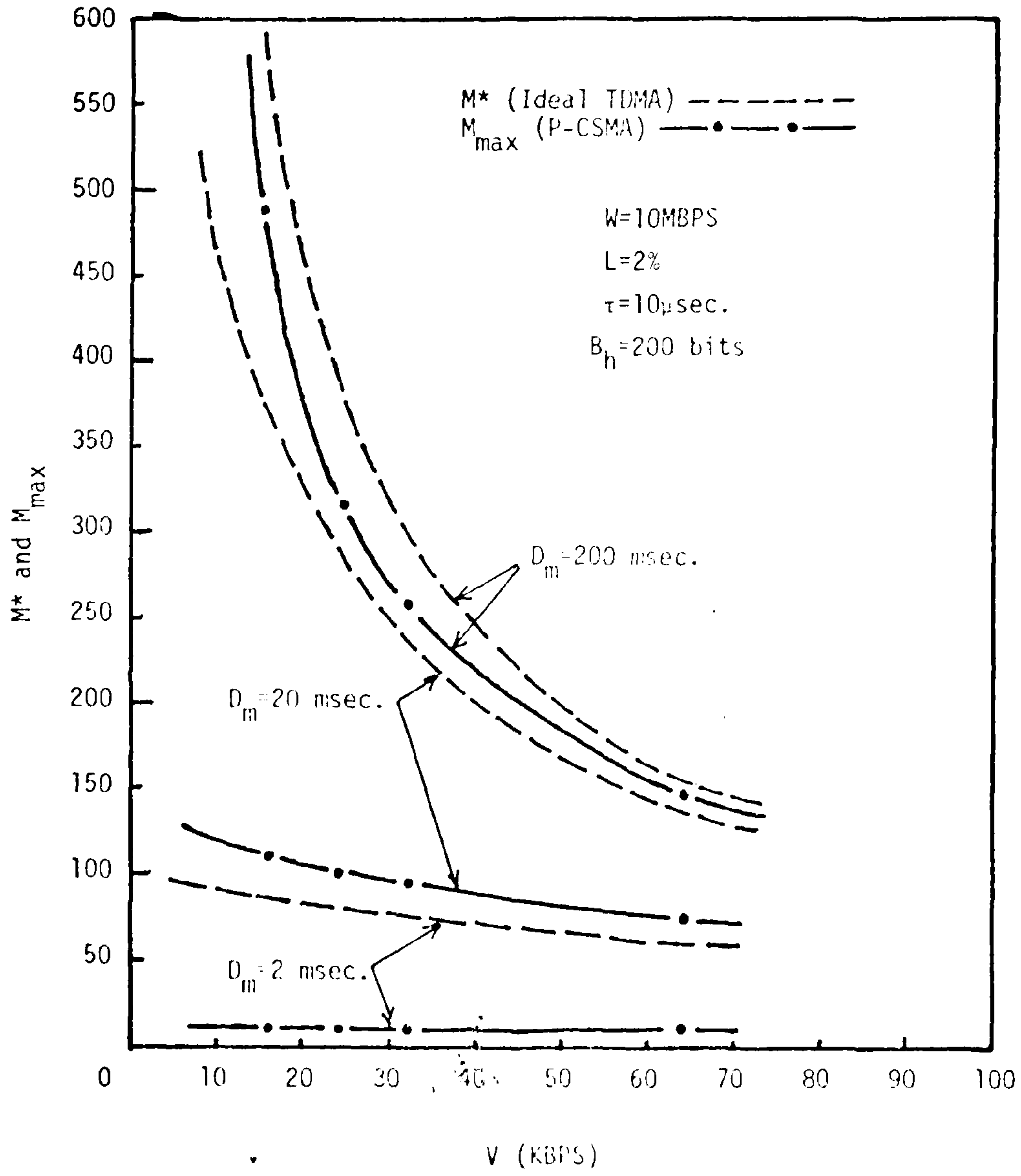


Figure 14 M_{max} and M^* versus V for W = 10 MBPS

the relative performance of P-CSMA when supporting voice. We plot in Figures 45 and 46 M_{max}/M^* versus V for $W = 1$ and 10 MBPS respectively. We note that the degradation in performance due to contention is more significant as the delay requirement is more severe and/or as the channel bandwidth is larger. Both these trends are due to the higher degree of contention caused by a larger ratio $\tau W/B$. Indeed, with smaller D_m, B_v is bound to be smaller and therefore $\tau W/B$ is larger. Finally in Figure 47 we display the optimum packet size B_v as a function of V for $L = 0.02$ and $D_m = 2$, 20 and 200 msec.

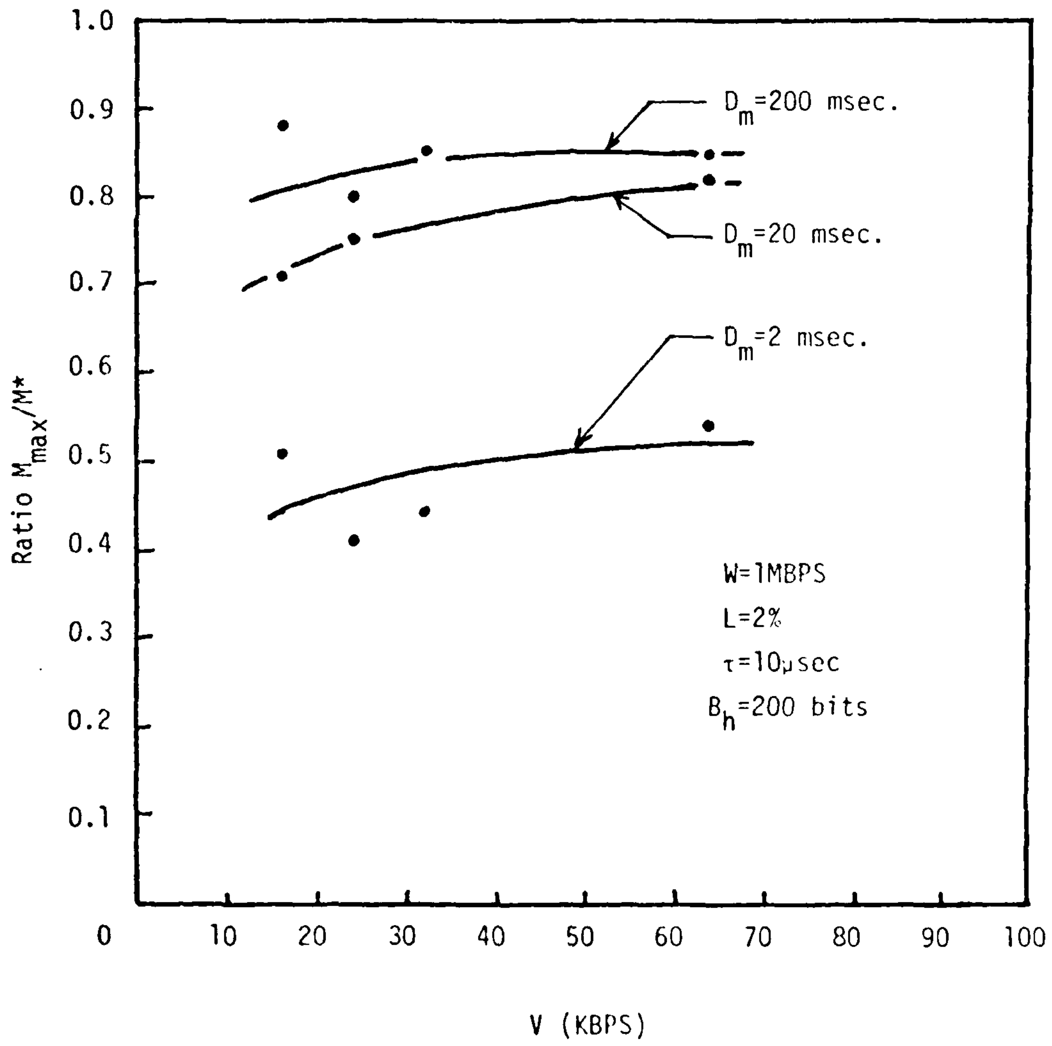


Figure 45 Ratio M_{max}/M^* versus V for W = 1 MBPS

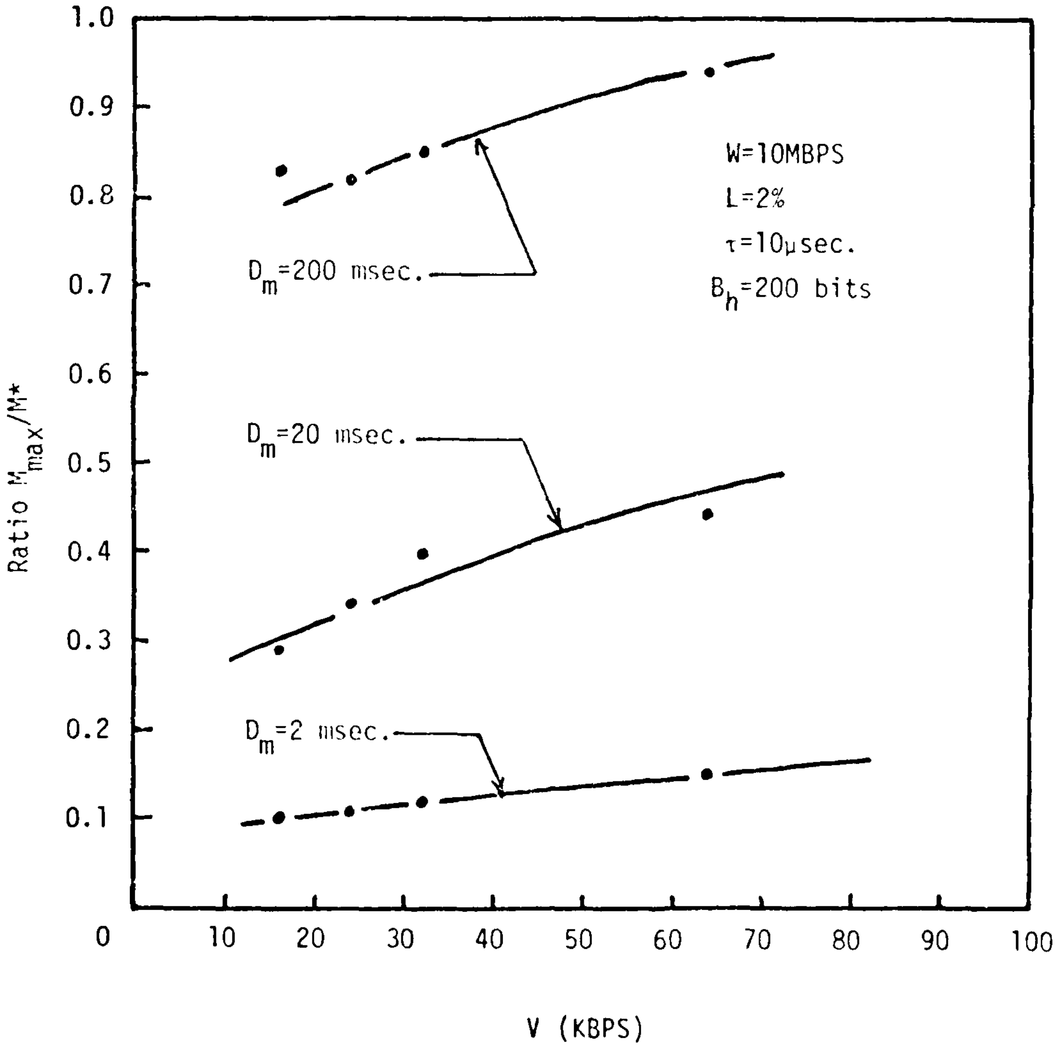


Figure 46. Ratio M_{max}/M^* versus V for $W = 10$ MBPS

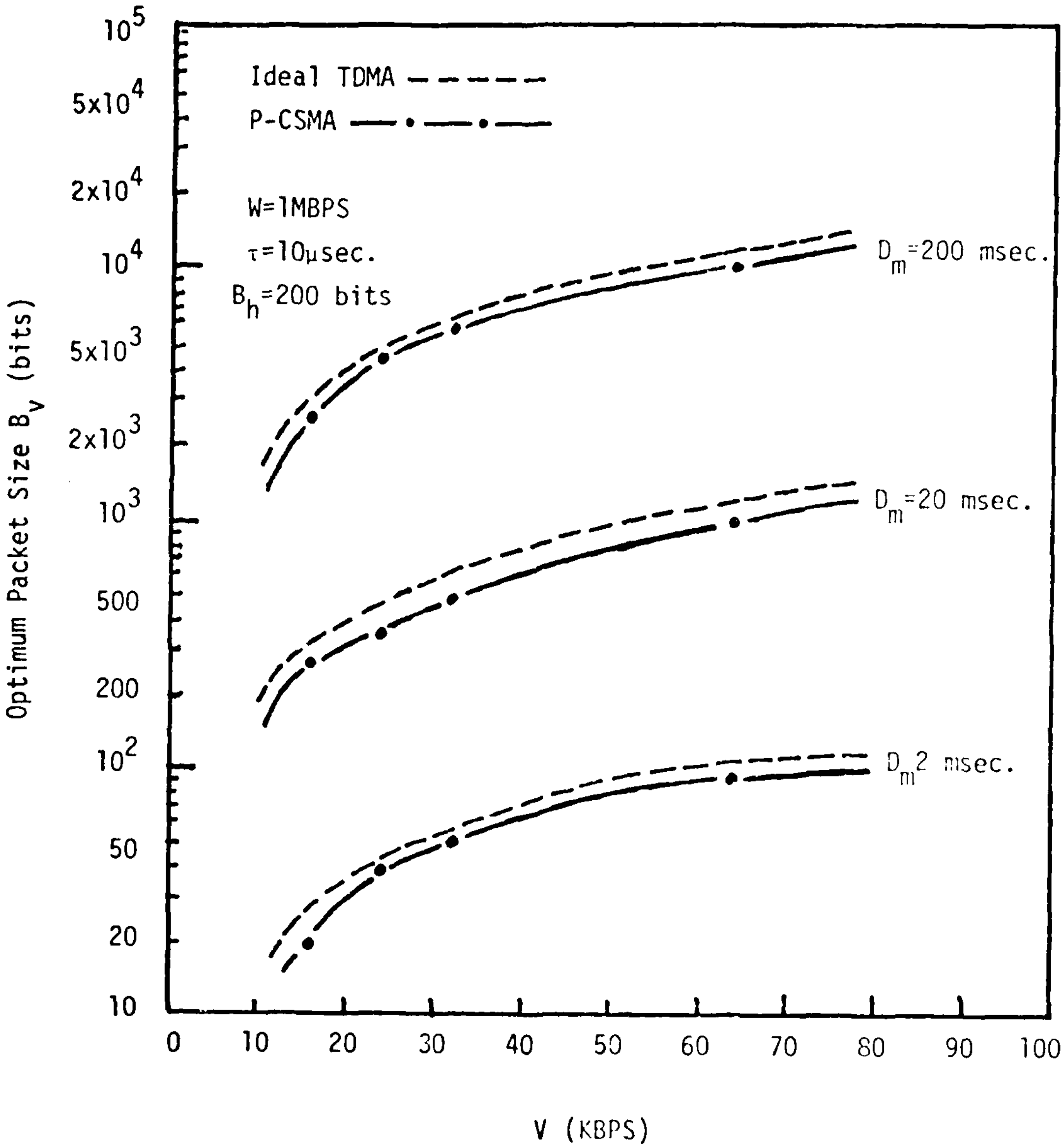


Figure 47. Optimum B_v versus V for W = 1 MBPS

§7 Conclusions and Suggestions for Future Work

The need for priority functions in Multiaccess Computer Communication Networks arises from applications in which we have restrictions on the delay that packets can incur. One alternative is to have a dedicated channel supporting very low loads to insure small delays; another alternative is to provide some priority mechanism by which packets that are time constrained can have priority over those which are not.

One such scheme, called P-CSMA was studied here using simulation. Three modes of operation were evaluated, namely: nonpreemptive, semipreemptive and preemptive. Generally, it was considered that the load placed on the channel by high priority packets was small ($S_1 = 0.1$). It was shown that priority functions indeed reduce packet delay while not reducing drastically the channel capacity in nonpreemptive and semipreemptive disciplines while the preemptive discipline offers excellent delay characteristics at the expense of reduced channel capacity due to thrashing on the channel. It was also shown that P-CSMA is not too sensitive to the collision detection feature offering similar performance even if this feature does not exist, making it useful for radio environments.

Packet loss for the high priority class is smaller than that in nonprioritized systems. More importantly, the results obtained show that C_1 loss does not grow toward one as the throughput approaches channel capacity. The addition of one extra buffer in the system does not affect average C_1-packet delay greatly nor its variance . But the real advantage is the reduction of packet loss in the system for both priority classes.

The variance of delay, a very important aspect in applications with real time constraints, was shown to be bounded and very stable for C_1-messages; preemption achieves a substantial reduction of variance as the load increases. This result insures small delays for C_1-messages regardless of S_2.

The performance of CSMA-CD local networks when supporting voice applications was examined and compared to an ideal TDMA system which provides the ultimate best achievable performance. The results show that for a given delay constraint D_m and a given tolerable loss rate L, *there is an optimum* packet size B_o which provides a maximum

number of voice sources. As long as the delay requirement is not too severe ($\approx$ 200 msecs.) and the channel bandwidth is not too large ($\approx$ 1 MBPS.) then the performance of P-CSMA is comparable to that of ideal TDMA. However if either D_m is small ($<$ 20 msecs.) or W is large ($\geq$ 10 MBPS.), or both then P-CSMA becomes inferior to the ideal case regardless of the vocoder rate. This is basically due to the relatively small transmission time of a packet for which P-CSMA is known to have a poor performance. As a result, we note that, when the delay requirement is low, an increase in channel bandwidth with the expectation to increase the maximim number of voice sources is rewarded by smaller than a proportional improvement.

Generally, priority functions are seen to accomplish the features they were designed for, namely: a reduction of packet delay, variance and loss for the high priority class. At the same time they offer an alternative in applications where time constraints force us to have a dedicated channel to insure small delays. Having a dedicated channel generally means a significant waste of channel capacity. Priority functions allow us to recover some of the wasted capacity without too much detriment to the performance of high priority classes. The scheme was shown to be very robust, as very little information is needed to perform the protocol. The separation of priority classes prevents the degradation of the performace of high priority classes as long as their load is kept below the channel capacity, independent of the load exercised by lower classes.

Throughout the course of this work a maximum of two priority classes were considered. There is still work to be done in evaluating the performance of P-CSMA with a larger number of classes and in the study of different mechanisms for priority assessment. With a larger number of classes, the priority assessment period of the protocol will play a more significant role in the average packet delay.

In the area of voice transmission there is also more work that can be done, finding models that incorporate silent periods, talkspurts,etc. These models can then be applied to the simulator on which this work is based.

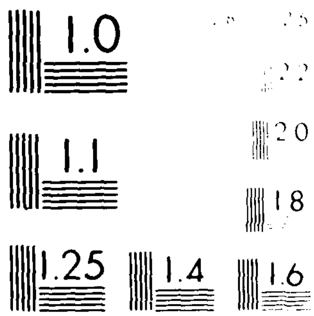
Another interesting aspect could be the study of ways to dynamically estimate

the number of users of a given priority class and adjust p accordingly to reduce the length of the contention period and increase the probability of a successful transmission.

AD-A112 564 STANFORD UNIV CA COMPUTER SYSTEMS LAB F/G 17/2
SIMULATION OF MESSAGE-BASED PRIORITY FUNCTIONS IN CARRIER SENSE--ETC(U)
JUN 81 N GONZALEZ-CAWLEY, F A TOBAGI DAAG29-79-C-0138
UNCLASSIFIED TR-213 ARO-16783.2-A-EL NL

2 of 2
AD A
112564

END
DATE
FILMED
04-82
DTIC


1.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

References

[1] N. Abramson, "The ALOHA System - Another Alternative for Computer Communications", AFIPS Conference Proceedings, 1970 Fall Joint Computer Conference.

[2] D. Clark, K. Pogran and D. Reed, "An Introduction to Local Area Networks" IEEE Proceedings, November 1978, pp. 1497-1517.

[3] L. Kleinrock and F. A. Tobagi, "Packet Switching in Radio Channels: Part I- Carrier Sense Miltiple Access Models and their Throughput Delay Characteristics.", IEEE Transactions on Communications, Vol. COM-25, Dec. 1975.

[4] D.E. Knuth, The Art of Computer Programming Vol. I: Fundamental Algorithms, Addison-Wesley , 1973.

[5] D.C. Loomis,"Ring Communication Protocols", University of California at Irvine, Department of Information and Computer Science, Technical Report #26, January 1973.

[6] R.M. Metcalfe and D.R.Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks", CACM, Vol.19, n.7, July 1976.

[7] J.R. Pierce, "Network for Block Switching of Data", Bell Systems Technical Journal, Vol. 51, July/August 1972.

[8] F.A. Tobagi and V.B. Hunt, "Performance Analysis of Carrier Sense Multiple Access with Collision Detection", Stanford Electronics Laboratories, Technical Report #173; June 1979.

[9] F.A. Tobagi, "Multiaccess Protocols in Packet Communication Systems", IEEE Transactions on Communications, Vol. COM-28, No. 4, April 1980.

[10] F.A. Tobagi, "Message-Based Priority Functions in Multiaccess/Broadcast Communication Systems with a Carrier Sense Capability", Proceedings of the Pacific Telecommunication Conference, Honolulu, Hawaii,Jan. 1980.

[11] F.A. Tobagi, "Performance Analysis of Carrier Sense Multiple Access with Message-

Based Priority Functions", Proceedings of the National Telecommunications Conference, Houston Tex., Dec. 1980.

[12] F.A. Tobagi, "Carrier Sense Multiple Access with Message-Based Priority Functions", Stanford Electronics Laboratories, Technical Report #200; Dec. 1980.

ATE

LMED

-8